Rendu Non-Photoréaliste à base de Textures Volumiques

Pierre Bénard

▶ **To cite this version:**

Pierre Bénard. Rendu Non-Photoréaliste à base de Textures Volumiques. Synthèse d'image et réalité virtuelle [cs.GR]. 2008. inria-00598456

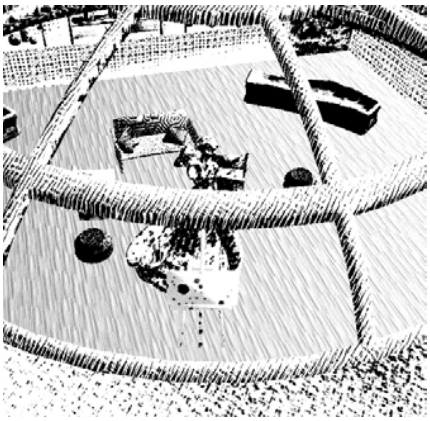
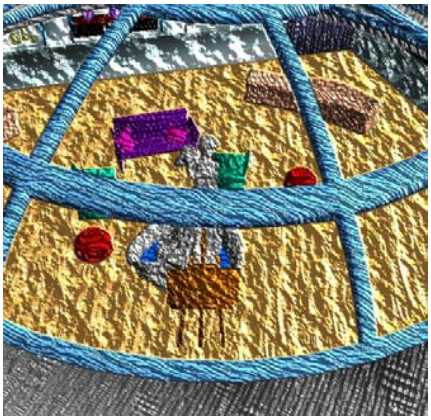
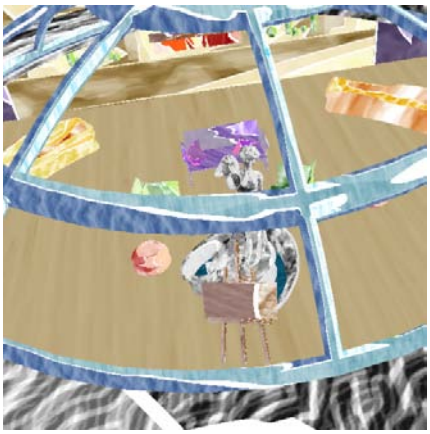
HAL Id: inria-00598456
https://inria.hal.science/inria-00598456

Submitted on 6 Jun 2011

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Rendu Non-Photoréaliste à base de Textures Volumiques

Master 2 Recherche Informatique
Promotion 2007/2008, Février-Juin 2008

Pierre BÉNARD
Superviseurs : Adrien BOUSSEAU et Joëlle THOLLOT
ARTIS - Laboratoire LJK
INRIA Rhône-Alpes - ZIRST
655 avenue de l'Europe, 38334 Saint Ismier Cedex

Remerciements

Je tiens à remercier Adrien Bousseau et Joëlle Thollot de m'avoir proposé ce si vaste et captivant sujet de Master, qui m'a permis de découvrir de nombreux domaines de la synthèse d'image et du rendu non-photoréaliste. J'aimerais également remercier Pascal Barla de m'avoir transmis son intarissable passion pour la recherche en général et ce domaine en particulier. Mes remerciements vont enfin à toute l'équipe Artis pour son accueil, et tout spécialement Pierre-Édouard et Thierry, qui ont eu la pénible tâche de partager leur bureau avec moi.

Abstract

This report presents a new solution to the issue of the temporal coherence of a non-photorealistic style during the animation of a 3D scene. This solution used “infinitly zoomable” solid textures in order to depict the medium of the stylization. After a detailed study of previous approaches to generate this 2D illusion of infinit zoom, new methods are described in the bi- and tridimensional cases. Finally various NPR styles are proposed to illustrate the quality, efficiency and simplicity of this approach.

Keywords : *Non-photorealistic rendering, temporal coherence, texture analysis and synthesis, infinite zoom.*

Résumé

Ce rapport présente une nouvelle solution au problème de la cohérence temporelle d’un style non-photoréaliste lors de l’animation d’une scène 3D. Cette solution se base sur l’utilisation de textures volumiques « infiniment zoomables » pour représenter le médium de la stylisation. Après une étude détaillée des approches précédemment proposées pour générer cette illusion de zoom infini en 2D, de nouvelles méthodes sont décrites dans les cas bi- et tridimensionnel. Enfin, différents styles NPR sont proposés pour illustrer la qualité, rapidité et facilité d’utilisation de cette approche.

Mots-clés : *rendu non-photoréaliste, cohérence temporelle, analyse et synthèse de texture, zoom infini.*

Table des matières

Chapitre 1

Introduction

Alors que la quête du réalisme avait motivé la recherche en informatique graphique depuis sa naissance, un courant parallèle, baptisé rendu non-photoréaliste (NPR), est apparu à la fin des années 80. Il s'inspire largement des techniques d'illustration traditionnelle – peinture, dessin au trait, mosaïque… – proposant non seulement des méthodes de simulation de ces styles mais surtout leur contrôle précis et leur animation. Pour une définition plus précise du NPR, nous vous invitons à consulter les ouvrages de Gooch et Gooch [28] et de Strothotte et Schlechtweg [84].

Le problème de la cohérence temporelle en NPR

Le travail présenté dans ce mémoire s'inscrit dans cette thématique. Son objectif est de proposer de nouvelles solutions au problème récurrent de la cohérence spatiale et temporelle du style au cours d'une animation NPR. Il est courant de séparer la stylisation d'une image ou d'une scène 3D en deux étapes, en distingant la stylisation de leurs régions de celle de leurs contours. En effet, un dessin est constitué d'un ensemble de contours – silouhette des objets, traits d'un visage…– qui définissent des régions, généralement fermées, chacune d'entre-elles possédant potentiellement un contenu différent (aplats de couleur, griffonage, hachures…). Nous appelerons ce contenu *remplissage* d'une région. Dans le cadre de cette étude, nous nous intéresserons uniquement à la cohérence de ce *remplissage* au cours d'une animation.

Produire une animation stylisée en deux dimensions à partir d'une scène 3D implique inévitablement l'apparition d'artefacts visuels. En effet, pour être parfaite, la stylisation d'une animation devrait satisfaire simultanément trois contraintes intrinsèquement contradictoires : le respect du mouvement 2D induit par celui 3D des objets de la scène, le respect des caractéristiques 2D (forme, fréquence, distribution…) des éléments du médium – pigments, hachures, points… – utilisé pour styliser le remplissage et la continuité au cours du temps de ce médium.

Sans traitement particulier, l'application des techniques de rendu non-photoréaliste à une succession d'images ne respecte qu'un sous-ensemble de ces contraintes et fait apparaître l'un des trois problèmes suivants. Soit l'effet de stylisation ne suit pas correctement le mouvement des objets, voire reste spatialement fixe tout au long de l'animation, créant un artefact appelé « rideau de douche » – l'observateur ayant l'impression de voir la scène à travers une feuille semi-transparente portant les éléments de style. Soit le médium portant la stylisation est déformé pour suivre le mouvement, s'étirant ou se concentrant en certaines parties de l'image, ce qui peut modifier profondément ses caractéristiques 2D. Soit, enfin, cet effet est appliqué sans cohérence à chaque image, pouvant varier en intensité et position aléatoirement (clignotements ou "popping") au cours de l'animation. Dans tous les cas, les perturbations introduites dégradent notablement la qualité de l'animation.

Le double problème du remplissage et de sa cohérence temporelle est omniprésent dans l'animation traditionnelle. En effet le remplissage utilisé par de nombreux styles – aquarelle, peinture, hachures… – est très fastidieux, voire impossible, à produire manuellement de façon cohérente tout au long d'un

dessin animé. Par conséquent, ces styles sont rarement utilisés en animation traditionnelle, au profit de remplissages uniformes.

Vers une solution à ce problème

Une animation non-photoréaliste peut faire intervenir deux types de mouvements : celui de la caméra observant la scène 3D et ceux des objets 3D faisant eux-mêmes partie de la scène. Notre problème est d'assurer que le remplissage de ces objets reste spatialement et temporellement cohérent tout au long de l'animation, quel que soit le type de mouvement. Cela implique, en particulier, que le médium suive les objets en mouvement et qu'il conserve des caractéristiques 2D quasi constantes à l'écran. La question de la cohérence du remplissage au cours du grossissement de l'objet, ou symétriquement du zoom de la caméra, est particulièrement épineuse. En effet, les marques de stylisation doivent conserver une échelle globalement invariante – car elles s'efforcent de représenter un médium 2D réel –, tout en traduisant l'effet de rapprochement par rapport à l'écran.

Ces deux contraintes sont intrinsèquement contradictoires. Il s'agit donc de trouver un compromis qui soit suffisamment continu pour assurer la cohérence temporelle de l'animation et procure une illusion de zoom suffisante pour tromper la vision de l'observateur. Ce paradoxe est encore renforcé par le fait que l'on souhaite que le zoom soit infini, afin de ne limiter – contrairement à la réalité – ni le mouvement de la caméra, ni celui des objets 3D.

L'informatique devrait être en mesure d'apporter une solution à ce problème nécessitant répétabilité et précision. Bien que de nombreuses approches aient été proposées (cf. chapitre 2), aucune d'entre-elles ne nous semble pleinement satisfaisante. Très complexes à mettre en œuvre, limitées à un nombre restreint de styles, les solutions actuelles ne nous paraissent pas assez simples et flexibles pour répondre aux attentes des utilisateurs – avant tout, des artistes.

Vue d'ensemble de notre approche

La contribution apportée par notre travail est double. Dans un premier temps, nous présenterons une étude détaillée du zoom infini bidimensionnel tel qu'il est décrit par Cunzi *et al.* dans *Dynamic Canvas for Immersive Non-Photorealistic Walkthroughs* [14] en fonction de critères de qualité statiques et dynamiques que nous définirons et d'une classification des textures appropriée à notre problème (cf. chapitre 3). Nous proposerons alors plusieurs améliorations de cette approche pour la rendre compatible avec une plus grande gamme de textures (cf. chapitre 4).

Dans un second temps, nous étendrons cette approche bidimensionnelle au cas d'une scène 3D dynamique, autorisant ainsi, à la fois les mouvements de la caméra et ceux des objets (cf. chapitre 5). L'idée générale de notre méthode est de représenter le médium par une texture volumique « infiniment zoomable » dans laquelle est plongé chaque objet animé. Ainsi, notre solution résout le problème de la cohérence spatiale – dans les trois dimensions, grâce au zoom infini – et temporelle.

Notre méthode apporte une solution simple d'utilisation. En effet, les textures volumiques ne nécessitent pas de paramétrisation : à tout point $P(x,y,z)$ de la surface d'un objet 3D, il est possible d'associer de façon triviale une couleur $T(x,y,z)$ dans la texture volumique, l'objet étant comme « sculpté » dans un bloc de matière. Les textures volumiques sont, par ailleurs, déjà intégrées dans de nombreux logiciels commerciaux de modélisation 3D (Maya©, 3ds Max©, Blender©...) et sont donc déjà connues des infographistes.

Enfin, avec notre approche, les styles de remplissage possibles ne sont limités que par la diversité des textures 2D ou 3D pouvant être synthétisées. Les récentes avancées en synthèse de texture volumique (cf. annexe A) permettent justement d'obtenir simplement une large galerie de motifs et matériaux. Nous avons ainsi pu adapter trois méthodes de stylisation existantes – aquarelle, style binaire et peinture – et en avons proposé une quatrième – le collage – utilisant notre zoom infini sur des textures 2D et/ou volumiques (cf. chapitre 6).

Chapitre 2

Cohérence temporelle : état de l'art

De nombreuses techniques de rendu non-photoréaliste d'une image fixe (dessin au trait, peinture, aquarelle, pointillisme, hachures…) ont été proposées depuis la fin des années 80. Les ouvrages de Gooch et Gooch [28] et de Strothotte et Schlechtweg [84] en donnent de nombreux exemples. Cependant une difficulté récurrente apparaît lors de l'extension de ces méthodes du cas statique à celui d'une animation : le problème de la cohérence temporelle.

Deux grandes classes d'approches ont été développées pour remédier à ce problème : la distribution de « primitives » – points, coups de pinceau, hachures… – à la surface des objets animés (section 2.1) et l'utilisation de textures liées aux objets animés, ou déformées pour suivre l'animation (section 2.2).

2.1 Approches par distribution de « primitives »

Être capable de générer une distribution de points de bonne qualité est indispensable pour de nombreux styles non-photoréalistes. Ces distributions servent en effet de points d'ancrage pour les marques de stylisation, qu'il s'agisse de coups de pinceaux [93], de fibres de papier [40] ou même de petits éléments de textures comme les *geograftals* [41].

La difficulté est non seulement d'obtenir une distribution statique satisfaisante mais encore de maintenir les qualités de cette distribution au cours du temps : celle-ci devant idéalement rester cohérente, en gérant les possibles apparitions/disparitions de points d'ancrage ou les occultations entre objets.

2.1.1 Distributions statiques

Obtenir une distribution de points statiques est directement lié au problème d'échantillonnage. En effet, il s'agit de déterminer le nombre et la position des points d'ancrage permettant un recouvrement de l'objet par des primitives 2D qui soit complet, non-uniforme – *i.e.,* fonction d'une carte d'importance, considérant plus de points dans les régions sombres d'une image, par exemple – et ne présente pas d'artefacts (régularité, alignements, motifs…).

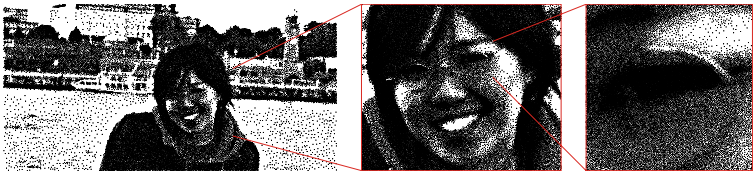
Satisfaire simultanément ces trois contraintes est un problème complexe, même pour une distribution statique. C'est pourquoi les approches dynamiques initiales, comme celle de Meier [63], ont laissé en suspens ce problème en utilisant une distribution complètement aléatoire de points d'ancrage, sans garantie de qualité.

Les premières solutions à ce problème en NPR trouvent leurs origines dans les techniques de demi-ton (*halftoning*) développées pour l'impression. En effet, les livres et journaux sont traditionnellement imprimés avec une seule encre – le noir –, mais comportent pourtant des images en niveaux de gris. Le processus de transformation de ces images continues en images binaires passe par le calcul d'une distribution de gouttelettes d'encre reproduisant fidèlement le ton original.

Ulichney [90] compare différents algorithmes répondant à ce problème et introduit les distributions ayant la propriété de *bruit bleu*, plaçant les points aléatoirement mais avec une contrainte minimale d'es-

pacement des voisinages. Les configurations de *bruit bleu* ont les avantages de l'apériodicité et de la structure non corrélée – comme les distributions aléatoires – mais sans la granularité des basses fréquences, ce qui les rend visuellement plus agréables. Il s'agit ainsi d'une distribution bien adaptée pour le tramage, les cellules rétiniennes étant naturellement disposées selon une configuration de *bruit bleu*.

(a) Rendu stippling par la méthode de Secord (81)



(b) Rendu stippling par la méthode de Kopf *et al.* (44). Noter la distribution non-uniforme des points adaptée au niveau de zoom.

FIG. 2.1: Deux méthodes de rendu stippling.

Le même type de problème se pose dans le cadre du rendu par points, ou *stippling*. Les algorithmes proposés pour le *halftoning* ont été adaptés dans ce nouveau cadre pour générer des distributions de base. Ces distributions initiales sont ensuite raffinées par relaxation, en suivant les interactions de l'utilisateur [17] (dessin à l'aide de brosses pour modifier localement la distribution) ou uniquement à partir de l'image à reproduire [81].

Les distributions ainsi obtenues sont de très bonne qualité (fig. 2.1(a)), possédant un spectre proche du *bruit bleu*. Ces méthodes sont cependant très coûteuses en temps de calcul – en particulier pour générer la distribution initiale.

Plus récemment, plusieurs approches [44, 49] ont permis d'obtenir en **temps réel** des distributions possédant des caractéristiques proches de celles d'un *bruit bleu*. Elles précalculent pour cela un ensemble de tuiles – petites images carrées – dont la combinaison, en respectant des règles d'adjacence, produit toujours de telles distributions. Parmi ces approches, l'intérêt particulier de celle proposée par Kopf *et al.* [44] réside dans le fait qu'un schéma de subdivision de ces tuiles permet de zoomer indéfiniement sur la distribution en maintenant ses bonnes propriétés (fig. 2.1(b)). Cette méthode propose ainsi un début de solution au problème des distributions de points dynamiques, malheureusement limitée au cas des images 2D.

2.1.2 Distributions dynamiques

FIG. 2.2: Rendu peinture par la méthode de Meier (63)

L'utilisation de distributions dynamiques de primitives a été introduite en NPR par Meier dès 1996 dans *Painterly rendering for animation* [63] (fig. 2.2). Le principe général de sa méthode est d'accrocher à la surface des objets 3D animés des points d'ancrage indiquant le placement des primitives 2D à rendre en espace image.

Cette approche a été abondamment étendue [13, 65, 68], notamment pour tenir compte du point de vue ou de l'importance relative des éléments de la scène. En effet, lorsqu'un objet s'éloigne de la caméra, la densité de la distribution de points lui étant associée devrait diminuer, la surface visible à l'écran de l'objet étant réduite. Ce comportement est obtenu par ces méthodes en précalculant une hiérarchie de points d'accroche « coupée » interactivement durant l'animation à la hauteur reproduisant au mieux la distribution souhaitée.

Si ces solutions assurent une très bonne cohérence lors du mouvement des objets, elles sont gourmandes en temps de calcul, ne gèrent pas les superpositions d'objets et ne peuvent pas garantir, dans tous les cas, une distribution des primitives ayant les propriétés d'un *bruit bleu*.

La connaissance de la scène 3D sous-jacente à l'animation n'est pas toujours disponible, notamment lorsque la source à styliser est une vidéo. Des approches à base de points d'ancrage ont néanmoins été

développées, ceux-ci suivant le flot optique[1] de la vidéo [31, 58], ou par différence entre image source et stylisée [36]. Ces approches souffrent cependant de deux limitations : elles ne permettent pas la gestion des occlusions entre objets – un point d'ancrage pouvant facilement passer d'un objet à l'autre – et elles ne sont adaptées qu'à un nombre restreint de styles.

Le meilleur compromis entre cohérence temporelle et qualité de la distribution dynamique semble, à l'heure actuelle, l'approche proposée par Vanderhaeghe *et al.* [92]. Il s'agit d'une technique hybride tirant partie de la richesse de l'information de mouvement des méthodes en espace objet et de la qualité des distributions 2D.

Pour y parvenir, ils génèrent une distribution statique bidimentionnelle de points pour la première image de l'animation. Ces points sont alors rétro-projetés, depuis l'espace image, à la surface des objets 3D. L'animation peut alors avancer d'une image, les points suivant le mouvement des objets. Les points déplacés sont ensuite reprojetés en espace image et la nouvelle distribution 2D ainsi obtenue est corrigée – par ajout/suppression de points – afin qu'elle conserve ses bonnes propriétés. Ainsi, la majorité des points est conservée d'une image à l'autre.

Pour augmenter encore la cohérence temporelle, non seulement l'apparition/disparition des points (et des primitives associées) est adoucie par un fondu sur plusieurs images, mais ces points sont également autorisés à glisser d'une zone saturée vers une zone en carence de points pour limiter au maximum l'effet de "popping". La figure 2.3 présente quelques images obtenues avec leur méthode dans différents styles non-photoréalistes.

(a) Pointillisme (b) Stippling (c) Peinture

FIG. 2.3: Différents rendus obtenus avec la distribution de Vanderhaeghe *et al.* (92).

La principale limitation de ces approches par distribution de primitive est qu'elles sont difficilement applicables dans le cas de styles « continus » ou par aplats (comme l'aquarelle) lorsqu'une primitive 2D est difficilement définissable. De plus, gérer de telles distributions et afficher les primitives 2D associées est généralement complexe et coûteux en temps de calcul. Les approches à base de textures fournissent alors une alternative intéressante.

2.2 Approches basées textures

L'utilisation de textures 2D est une technique courante en informatique graphique pour enrichir simplement, et pour un coût limité, l'apparence des objets 3D, qu'il s'agisse de leur couleur, de leur réaction à la lumière voire de leur géométrie. Ces textures sont utilisées de deux façons différentes en NPR.

Généralement, elles sont plaquées en espace objet – dans le référentiel de la scène 3D – si bien qu'elles restent intrinsèquement cohérentes. La texture suivant les mouvements de l'objet auquel elle est accrochée, l'effet « rideau de douche » est ainsi évité. Il est cependant nécessaire de trouver une bonne paramétrisation de la texture pour que les déformations et discontinuités soient limitées lors du placage.

Mais les textures peuvent également être utilisées directement en espace image. En effet, pour générer une animation NPR cohérente en espace image, la texture portant les marques de stylisation doit satisfaire deux contraintes concurrentes : suivre le mouvement de l'animation et conserver son apparence générale en terme de distribution et de fréquence. L'idée générale de ces approches est qu'en utilisant

[1] Estimation du mouvement de chaque pixel entre deux images successives, sous les hypothèses d'une illumination constante et de l'absence d'occlusion

une information de mouvement extraite de la scène 3D ou d'une vidéo, les textures peuvent être déformées pour suivre l'animation. Le problème central est alors d'établir une correspondance correcte entre le mouvement 3D des objets animés et celui 2D de la texture.

2.2.1 Méthodes en espace objet

La cohérence intrinsèque apportée par les textures plaquées sur les objets 3D a été exploitée en NPR, notamment par Klein *et al.* [42] pour la visite d'environnements virtuels et Praun *et al.* [74] pour le rendu temps-réel de hachures. Deux difficultés majeures subsistaient cependant : la création du "mapping" entre l'objet et sa ou ses textures, ainsi que le contrôle de la stylisation lors des variations des conditions d'affichage (zoom, illumination...).

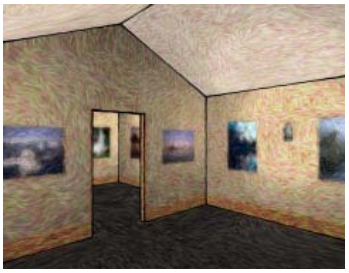


FIG. 2.4: Environnement virtuel NPR obtenu par la méthode de Klein (42)

Klein *et al.* proposent un système de rendu à base d'images (IBR) stylisées par des filtres non-photoréalistes. Ils bénéficient ainsi des avantages des deux techniques : des modèles 3D simples enrichis par la complexité des effets de lumière et des détails géométriques de l'IBR et la stylisation du NPR. En outre, ils résolvent le problème du contrôle de la taille des marques de stylisation à différents niveaux de zoom en construisant pour chaque texture son *art map*, pyramide d'images (*mip-map* ou *rip-map*) stylisées à différentes échelles. Lorsque le visiteur virtuel navigue interactivement dans un tel environnement, la texture utilisée est un mélange des deux images de l'*art map* les plus proches du facteur de zoom actuel. Ainsi les marques de stylisation ont une taille globalement constante à l'écran et le zoom est continu.

Deux limitations principales à cette approche sont à noter. Les filtres NPR étant appliqués de façon indépendante à chaque niveau de l'*art map*, il n'y a pas cohérence spatiale des effets tout au long de la pyramide. Même si le mélange de deux niveaux de cette pyramide lors du rendu limite ce problème, le zoom ne peut être parfaitement cohérent.

Praun *et al.* répondent à ce dernier problème en introduisant une séquence de *mip-map*, appelée *tonal art map* (TAM). Ces textures sont combinées à l'exécution, en fonction des conditions d'illumination, pour texturer les objets 3D. La cohérence temporelle des tons et de la résolution durant le zoom est assurée par construction de la TAM. En effet, les hachures d'une image claire forment un sous-ensemble de celles d'une image plus foncée (fig. 2.5 : de gauche à droite) et les hachures d'une résolution grossière apparaissent dans celles d'une résolution plus fine (fig. 2.5 : de haut en bas).

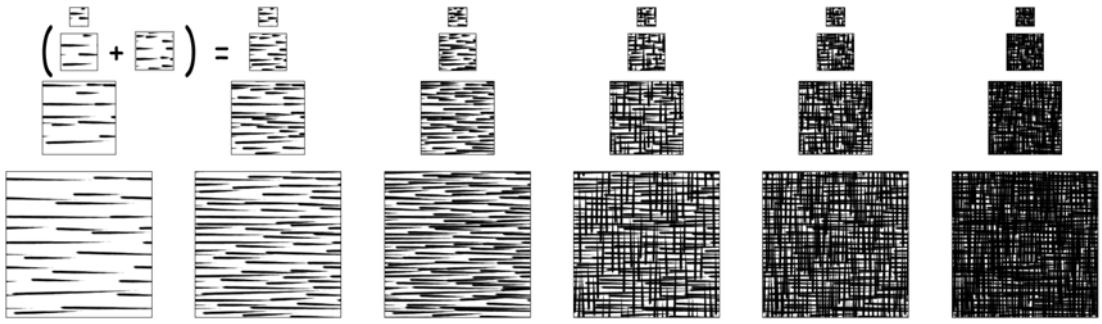


FIG. 2.5: *Tonal Art Map* de Praun *et al.* (74). Les hachures d'une image apparaissent dans toutes celles qui la suivent en dessous ou à sa droite.

Ainsi, lorsque les conditions d'affichage vont changer, comme plusieurs images de la TAM sont mélangées et partagent de nombreuses hachures, seules certaines marques vont disparaître/apparaître progressivement : la majorité d'entre-elles subsistent à la même position, évitant ainsi l'effet de clignotement.

La paramétrisation des éléments de textures reste cependant complexe, Praun *et al.* utilisant celle des *Lapped Textures* [73] (cf. section A.1.2 pour plus de détails concernant cette méthode). En outre, la création de la TAM devient difficile voire impossible pour d'autres marques que des hachures.

2.2.2 Méthodes entre espace objet et espace image

Plusieurs approches à mi-chemin entre espace objet et espace image ont été proposées : l'information de mouvement 3D étant extraite de la scène en espace objet et convertie en mouvement 2D d'une ou plusieurs textures.

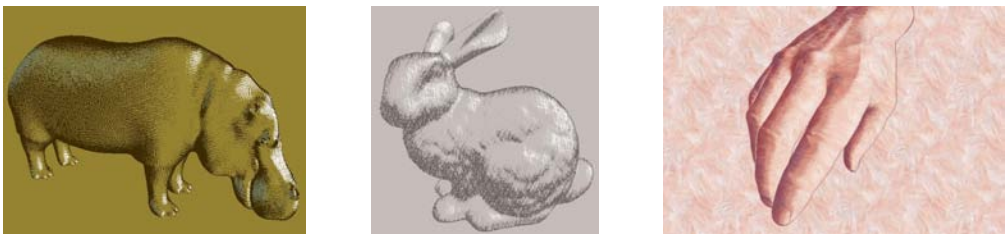
Cunzi *et al.* [14] proposent ainsi de faire subir une transformation géométrique – en espace image – inverse au mouvement de la caméra – en espace objet – à une texture d'arrière plan, servant de toile de fond à une visite virtuelle d'une scène 3D. De cette façon, le sentiment d'immersion du visiteur virtuel est augmenté, la texture de papier suivant ses déplacements (plus de détails section 3.1.2).

FIG. 2.6: Rendu "encre" de paysages par la méthode de Coconu *et al.* (11)

Cette approche est cependant restreinte à une seule texture d'arrière-plan pour toute la scène. En outre, la transformation étant calculée par rapport au mouvement de la caméra, un objet 3D se déplaçant dans la scène semble glisser sur le fond, produisant l'effet « rideau de douche », ce qui limite cette méthode aux scènes statiques.

Pour pallier cette limitation, Coconu *et al.* [11] pour le rendu "encre" de paysages (fig. 2.6) et Breslav *et al.* [9] pour le rendu à base de « motifs » (fig. 2.7) introduisent des méthodes basées sur le mouvement des objets 3D de la scène.

Ces deux méthodes consistent à calculer la transformation 2D de la texture combinant rotation, translation et mise à l'échelle – *i.e.,* la similitude – qui soit la plus proche du mouvement 3D des objets sous-jacents. Pour déterminer cette transformation, la projection dans l'espace image d'un seul (Coconu *et al.*) ou d'un ensemble de points 3D (Breslav *et al.*) est suivie image après image.



(a) Halftone (b) Hachures (c) Peinture

FIG. 2.7: Différents rendus obtenus avec la méthode de Breslav *et al.* (9).

L'intégralité d'un objet ne suivant pas nécessairement le même mouvement 2D, Coconu *et al.* groupent les éléments de la scène en « primitives de haut niveau » (HLP) tandis que Breslav *et al.* permettent à l'utilisateur de subdiviser la surface des objets 3D en "patches", texturés de façon indépendante. Même si le raccordement entre ces éléments est géré par mélange, des artefacts peuvent apparaître car la paramétrisation en bordure des HLP n'est pas complètement cohérente dans la méthode de Coconu *et al.* et le mélange dépend de la régularité du maillage de l'objet 3D dans l'algorithme de Breslav *et al.*.

Si ces approches produisent des résultats purement 2D – donc en accord avec la nature traditionnelle du motif – de bonne qualité, trouver une similitude 2D visuellement proche d'un mouvement 3D est un problème difficile, voire impossible dans certains cas (objet allongé tournant autour d'un axe parallèle au plan 2D, par exemple), si bien que le mouvement des textures n'est pas toujours cohérent.

2.2.3 Méthodes en espace image

Une dernière classe d'approches tente de résoudre le problème de la cohérence temporelle purement en espace image. Bousseau *et al.* [8] proposent ainsi de convertir une vidéo dans un style aquarelle (fig. 2.8) en utilisant l'advection de texture [66].

(a) Image originale. (b) Rendu aquarelle.

FIG. 2.8: Rendu aquarelle de vidéo d'après la méthode de Bousseau *et al.* (8).

Bousseau *et al.* [7] ont montré que des résultats en aquarelle visuellement convaincants peuvent être obtenus en modifiant une image couleur par l'utilisation de textures de pigments et de papier en niveaux de gris (pour plus de détails cf. section 6.1). Un moyen de transformer une vidéo dans un style aquarelle est donc de lui compositer une texture de papier/pigment. Cependant, cette texture ne doit pas rester fixe tout au long de l'animation – sans quoi l'effet « rideau de douche » est garanti – mais doit suivre les mouvements de l'animation. Bousseau *et al.* proposent d'utiliser l'advection de texture – traditionnellement utilisée pour représenter des flux en visualisation scientifique – pour répondre à ce problème.

L'advection de texture part d'une paramétrisation initiale de la texture à la première image de l'animation et modifie progressivement, pixel par pixel, cette relation en suivant le flot optique de la vidéo. La paramétrisation est réinitialisée périodiquement, lorsque l'apparence de la texture déformée diverge trop de l'original.

Cette méthode permet de capturer très précisément des mouvements complexes et de gérer les occlusions et désocclusions, dans la mesure où le flot optique (connu dans le cas d'une scène 3D ou extrait de la vidéo) est correct, ce qui est malheureusement loin d'être systématiquement le cas. Outre cette dépendance vis à vis du flot optique, une autre limitation de cette approche est qu'elle nécessite la connaissance de l'intégralité de l'animation (du début jusqu'à la fin) ce qui exclut toute utilisation interactive. Enfin, cette méthode ne peut utiliser qu'une seule texture pour toute la scène, ce qui restreint nettement la richesse de la stylisation.

Deux autres approches en espace image, prenant une vidéo en entrée mais proposant un remplissage hybride entre « primitives » et textures, sont encore à signaler. Les méthodes de Wang *et al.* [95] et Kolliopoulos *et al.* [43] sont basées sur une segmentation temporellement cohérente de la vidéo. Elles proposent une extension d'algorithmes classiques de segmentation 2D (*Mean Shift* [12] et *Normalized Cuts* [83], respectivement) à un « volume temporel » (2D + temps).

Les différentes marques de style – couleurs uniformes, coups de pinceau, points… – appliquées aux zones ainsi segmentées restent alors cohérentes durant l'animation. La qualité finale tient pour beaucoup, comme précédemment pour le flot optique, à celle de l'algorithme de segmentation. L'extension de ces méthodes pour une utilisation interactive semble en outre très difficile.

FIG. 2.9: Rendu peinture par la méthode de segmentation de Kolliopoulos *et al.* (43).

2.3 Bilan et conséquences

Bien que la cohérence temporelle constitue un problème majeur en rendu non-photoréaliste et que de nombreuses solutions aient été proposées tout au long de ces quinze dernières années, aucune véritable étude ciblée sur ce thème – proposant, par exemple, une mesure perceptuelle de ses effets – n'a été menée. Il est par conséquent difficile d'évaluer quantitativement la qualité d'une animation ou d'un algorithme de rendu NPR en termes de cohérence temporelle et donc de comparer strictement les différentes approches présentées précédemment.

Si l'objectif de notre travail n'est pas d'établir une telle mesure ou de réaliser ce type de comparaison – mais plutôt de proposer une solution originale à ce problème – nous nous efforcerons de garder à l'esprit cette problématique plus large, en prenant soin de justifier notre méthode par des critères aussi tangibles que possible.

Au regard des travaux précédents, il est possible de déterminer certaines pistes pour résoudre notre problème. Dans la mesure où nous souhaitons une solution simple et applicable à un grand nombre de styles NPR, les approches basées textures semblent à privilégier : elles ne nous restreignent en effet pas à des styles possédant des marques clairement distinctes. Plus précisément, les méthodes en espace objet semblent particulièrement intéressantes puisqu'elles résolvent en grande partie le problème de la cohérence spatiale et temporelle – la texture étant liée à l'objet 3D – tout en permettant l'utilisation de plusieurs textures pour une même scène, voire un même objet 3D. Ces approches laissent cependant deux problèmes en suspens : celui de la cohérence durant le zoom et de la paramétrisation des textures.

Pour nous affranchir de ce dernier problème inhérent aux textures 2D, nous proposons d'utiliser des textures volumiques qui ne nécessitent pas de paramétrisation. Libérés de ce problème, la seule question restante est celle du zoom. Afin de ne restreindre ni le mouvement de la caméra, ni celui des objets 3D, il est nécessaire que ce zoom soit infini et adapté à toutes marques de stylisation, donc tout type de texture. Nous proposons dans le chapitre suivant un ensemble de solutions permettant de créer cette illusion de zoom infini dans le cas de textures 2D, en étendant la méthode proposée par Cunzi *et al.*.

Chapitre 3

Étude du zoom infini bidimensionnel

Dans ce chapitre nous proposons une étude détaillée des méthodes existantes pour créer l'illusion d'un zoom infini sur une image bidimensionnelle. Après avoir déterminé les critères de qualité de cette illusion, nous étudierons, pour chaque classe de textures, la qualité des résultats obtenus par la principale approches capable de produire cette illusion : "Dynamic Canvas".

3.1 L'illusion du zoom infini

Exception faite des motifs fractales – qui, par définition, sont autosimilaires à un certain facteur d'échelle près –, il semble impossible de zoomer indéfiniment sur un image quelconque, d'autant plus en ne connaissant que son apparence à une échelle donnée. L'impression de zoom infini ne peut donc que résulter d'une illusion trompant les sens de l'observateur, comme les escaliers sans fin (fig. 3.1) du peintre M.C. Escher.

FIG. 3.1: *Ascending and Descending* by M.C. Escher (1960).

3.1.1 Une illusion sonore

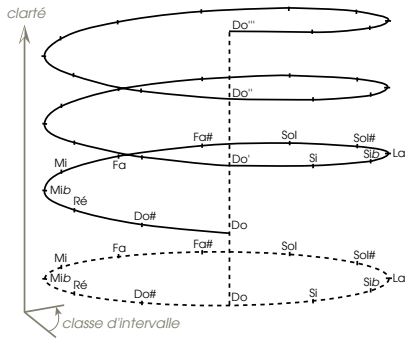


FIG. 3.2: Modèle en hélice de la hauteur d'un son.

En dimension 1 tout d'abord, pour un signal sonore, des travaux ont montré que l'illusion de l'infini pouvait être créée dans le cas de la perception de la *hauteur* d'un son, *i.e.,* directement liée à sa fréquence (inverse de sa vitesse de vibration). Normalement, plus un son est haut (resp. bas) plus il est perçu comme aigu (resp. grave).

La hauteur d'un son peut cependant être décomposée, d'après Drobisch [20], selon deux dimensions (fig. 3.2) : une *classe d'intervalle* – écart de hauteur entre deux notes (position sur le cercle) – et une *clarté* – quantité de hautes fréquences (axe vertical) –, formant ainsi un modèle en hélice.

Gamme de Shepard (82)

En 1964, l'objectif de R.N Shepard [82] est de générer des tonalités avec une *classe d'intervalle* différente mais la même *clarté*. Sur le modèle en hélice des hauteurs de la figure 3.2, cela consiste à ne pas se déplacer verticalement mais à tourner sur le cercle des classes de hauteur, *i.e.,* la projection de l'hélice.

Une *tonalité de Shepard* (fig. 3.3) est un son composé d'un certain nombre d'harmoniques – ondes de fréquences multiples de la fréquence fondamentale : 440 Hz pour un "La" – espacées d'une octave

(fréquence doublée) et modulées par une enveloppe gaussienne. La forme exacte de cette enveloppe n'est cependant pas cruciale, contrairement à la relation fréquentielle entre les harmoniques [75].

La *gamme de Shepard* est alors créée en déplaçant la tonalité précédente vers les fréquences croissantes d'un pas constant (en échelle logarithmique) en conservant l'enveloppe à la même position. Les Figures 3.3 (a) à (c) illustrent les déplacements successifs – la fréquence fondamentale étant discriminée en vert – qui permettent de réaliser la gamme. Lorsque le déplacement correspond à une octave, la tonalité initiale est retrouvée.

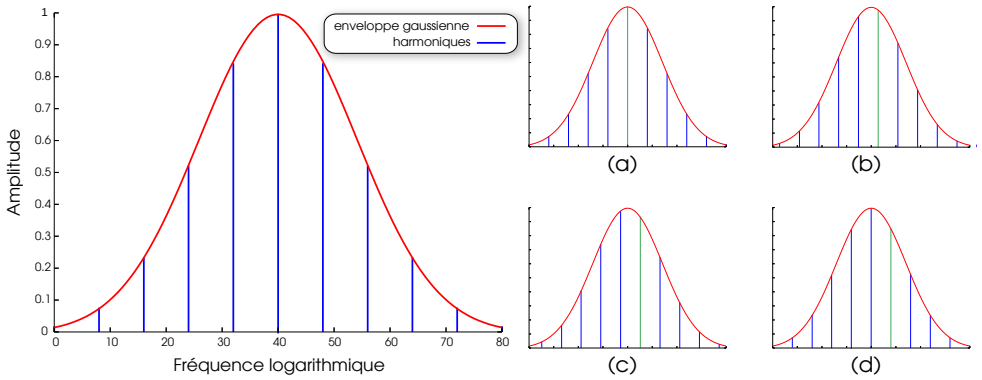


FIG. 3.3: Tonalité et gamme de Shepard.

L'illusion consiste alors à jouer en boucle cette gamme, sans répéter les deux tonalités équivalentes. L'oreille de l'auditeur a tendance à suivre la plus forte tonalité – celle au centre de l'enveloppe – dont la hauteur est croissante. Alors que l'intensité de celle-ci diminue, l'oreille est attirée par la précédente tonalité qui gagne en intensité et suit sa montée en hauteur. C'est ce passage inconscient d'une tonalité à l'autre qui crée l'illusion d'une gamme montant à l'infini.

Glissando de Risset (80)

Le compositeur et chercheur Jean-Claude Risset [80] étend la *gamme de Shepard* en une version continue intitulée *glissando en spirale*[1]. Il s'agit d'un agencement de douze sons complexes (synthétisés uniquement par ordinateur) composés de six notes de six octaves différentes. Pour que l'illusion fonctionne, il faut introduire ou éliminer de façon très graduelle les notes de fréquences graves et aiguës.

Risset utilisa en 1968 ce type d'illusion, lorsqu'il mit en musique le texte *Little Boy* du dramaturge Pierre Halet, pour symboliser la chute de la bombe atomique sur Hiroshima.

3.1.2 Une illusion visuelle

"Endless zoom" de Glassner (27)

Le passage d'un signal unidimensionnel – un son – à un signal bidimensionnel – une image – a été proposé par Andrew Glassner [27] en suivant strictement le même schéma que Shepard. Glassner propose de synthétiser une image à partir de fonctions périodiques, par exemple $A(x,y,f) = \sin(fax) * \cos(fby)$ avec a et b deux constantes. Partant d'une image blanche, il somme en chaque pixel la fonction A évaluée en $f/i\ \forall i \in \{-N \dots N\}$ pondérée par une enveloppe gaussienne.

Pour créer une animation cyclique de F images, il suffit d'utiliser pour chaque image i la fréquence fondamentale $f + (F/i)f$. Ainsi la fondamentale de la dernière image est $2f$ et le cycle est possible. Comme l'illustre la figure 3.4, Glassner constate que les images ne semblent pas grandir indéfiniment, de la même façon que monte le signal audio de Shepard. Il les décrit plutôt comme étant vues à travers un long tube dans lequel l'observateur volerait en avant ou en arrière à l'infini.

[1] La gamme de Shepard et le Glissando de Risset peuvent être écoutées sur : http://asa.aip.org/demo27.html

Cette méthode est cependant limitée à une image totalement procédurale, pouvant être générée à partir d'une somme de fonctions périodiques.

FIG. 3.4: "Endless zoom" de Glassner avec $A(x,y,f) = \sin(2f\pi x) * (\sin(\pi fy) + \cos(3\pi fy))$.

"Dynamic Canvas" (14)

Basée sur le même principe, la méthode proposée par Cunzi *et al.* dans *Dynamic Canvas for Immersive Non-Photorealistic Walkthroughs* [14] permet l'utilisation aussi bien d'images procédurales que réelles.

L'objectif de ce travail est d'accroître la sensation d'immersion d'un visiteur virtuel se promenant dans un environnement 3D statique, rendu de façon non-photoréaliste. L'idée est d'animer l'arrière-plan (texture de papier, toile...) de la scène en traduisant en 2D les mouvements 3D de la caméra : translations et rotations mais aussi zoom.

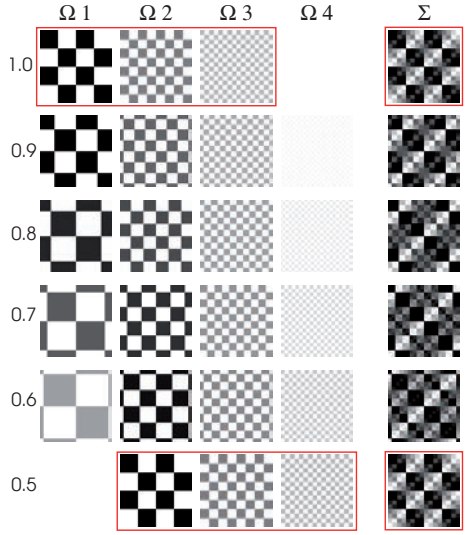


FIG. 3.5: Zoom infini de "Dynamic Canvas" sur une texture d'échiquier. Le facteur de zoom est montré à gauche.

Pour permettre cette impression de zoom infini, Cunzi *et al.* proposent d'utiliser quatre versions de la texture d'arrière-plan $\Omega_i \; \forall i \in \{1..4\}$ redimensionnées d'un facteur 2^{i-1} (cf. première ligne de la figure 3.5) : ce qui est analogue aux différentes octaves sonores de Shepard. La somme Σ, pondérée par une enveloppe linéaire, de ces quatre octaves correspond à la texture d'arrière-plan pour le facteur de zoom initial ($zoom = 1$).

Lorsque le visiteur avance dans le papier, le facteur de zoom diminue et toutes les octaves sont redimensionnées selon cette valeur. Lorsque $zoom = \frac{1}{2}$, il est possible de décaler toutes les octaves car $\Omega_{i+1}^{zoom=\frac{1}{2}} = \Omega_i^{zoom=1}$, la dernière octave Ω_4 étant réinitialisée et le facteur de zoom remis à 1.

La figure 3.6 montre un résultat obtenu avec cette méthode à partir d'une texture de papier scannée. Le mélange semble très satisfaisant pour ce type de texture et d'application, car le « motif » de la texture est peu discernable. Cependant dans le cas d'une texture possèdant un « motif » précis – comme l'échiquier –, on constate que celui-ci est plus difficilement reconnaissable dans la texture résultat, les octaves de différentes fréquences étant mélangés mais discernables.

Le zoom infini de "Dynamic Canvas" semble un intéressant point de départ dans la mesure où :

- il trouve sa justification dans les travaux précédemment menés en dimension 1,
- il produit des résultats satisfaisants pour certains types de textures.

La dégradation du « motif » qu'entraîne le mélange est cependant problématique pour certaines classes de textures. Il est donc nécessaire de déterminer les critères définissant la qualité de la texture résultat afin de cibler les aspects de la méthode qui sont à améliorer.

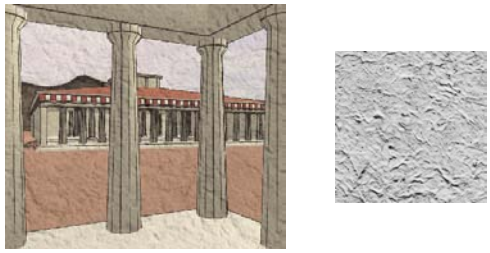


FIG. 3.6: Exemple de "Dynamic Canvas" sur une texture scannée (à droite).

Il est en outre important de rappeler les deux autres limitations de cette approche, qui seront levées au chapitre 5. Cette méthode n'est capable d'utiliser qu'une seule texture 2D par scène, tous les objets 3D étant « dessinés » sur ce fond, et ces objets sont nécessairement statiques, seule la caméra pouvant se déplacer dans l'environnement virtuel.

3.2 Critères de qualité

Comme souligné en conclusion de l'état de l'art sur la cohérence temporelle (cf. chapitre 2), aucune étude n'a défini à ce jour une mesure formelle de la cohérence d'une animation non-photoréaliste. Cette carence s'explique probablement par la difficulté que nous avons à expliciter les critères permettant de juger cette cohérence, essentiellement perceptuelle, et a fortiori d'en établir une mesure.

Nous essaierons cependant dans cette section de déterminer les principaux critères de qualité statiques et dynamiques, à prendre en compte dans le cas particulier du zoom infini bidimensionnel.

3.2.1 Critères statiques

Les critères statiques définissent les qualités souhaitées pour n'importe quelle image fixe I, extraite d'une séquence du zoom infini sur une texture T.

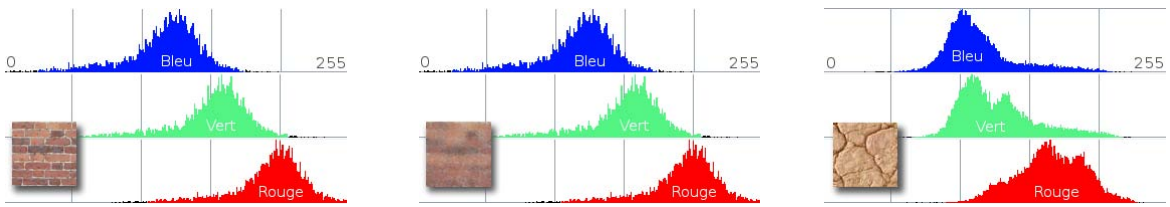
Similarité avec la texture d'origine

Étant donné que l'artiste choisit en entrée du zoom infini une texture T, le premier critère de qualité semble être la similarité de I avec T. Cette similarité peut être définie, selon nous, de trois façon différentes.

Il peut tout d'abord s'agir d'une simple ressemblance visuelle. Si ce critère est le plus simple à utiliser, car à la porté de chacun, il est cependant totalement subjectif et varie donc d'un observateur à l'autre. Une utilisation rigoureuse de ce critère nécessiterait la mise au point d'une expérimentation sur un panel significatif d'utilisateurs, ce que nous n'avons pas eu la possibilité de réaliser lors de cette étude. Pour autant, nous ne le négligerons pas, car il constitue le principal critère d'un spectateur lambda visionnant une animation non-photoréaliste.

Pour obtenir un critère plus facilement mesurable, il est possible de considérer la similarité de deux images comme leur proximité en termes de couleurs et d'utiliser pour cela leurs histogrammes (fig. 3.7). L'histogramme est un graphique statistique permettant de représenter, par canal de couleur, la distribution des intensités des pixels d'une image, *i.e.,* le nombre de pixels pour chaque intensité lumineuse. Ainsi comparer les deux triplets d'histogrammes de I et T revient à comparer la similarité des tons dominants des deux images.

Cependant, deux images aux couleurs semblables ne sont pas nécessairement similaires – les histogrammes de la figure 3.7(a) et 3.7(b) sont strictement identiques – la texture (b) étant une version aléatoirement brouillée de la texture (a) – sans pour autant que les textures se ressemblent. De même, les histogrammes 3.7(a) et 3.7(c) ne sont pas radicalement dissimilaires alors que les textures n'ont que peu de points communs.



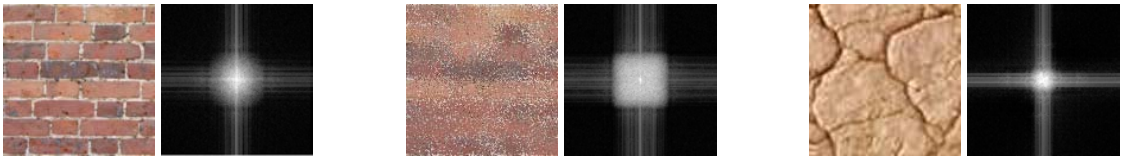
(a) Texture de briques (b) Texture précédente brouillée (c) Texture de sol

FIG. 3.7: Histogrammes de trois textures.

Les textures possèdent généralement un « motif », identifiable, voire segmentable. Dans ce cas, la similarité entre deux textures peut se définir en terme de distribution de ce « motif ». Extraire ce « motif » de la texture et mesurer cette distribution sont néanmoins des tâches difficiles qui dépassent le cadre de notre étude.

Une dernière information, plus facilement extractible d'une image, par transformée de Fourier rapide, est sa distribution fréquentielle. Il est alors envisageable de comparer I et T en fonction de leur spectre (module de la transformée de Fourier). Le spectre fournit, en effet, certaines caractéristiques de la texture. En particulier, les pics présents dans le spectre nous informent sur l'orientation de la structure ainsi que sur son amplitude. En effet, le pic dominant dans le spectre donne la direction principale de la texture, tandis que la localisation des pics permet de déterminer la période spatiale fondamentale de la texture. Enfin, chaque pic représente une périodicité spatiale dans un sens particulier.

Les spectres extraits pour les deux textures précédentes (assez proches en couleurs) sont bien distincts (fig. 3.8). Cependant, cette représentation est peu intuitive, assez difficile à interpréter et peu pertinente pour les textures ne présentant pas une certaine structure.



(a) Texture de briques (b) Texture précédente brouillée (c) Texture de sol

FIG. 3.8: Spectres (droite) de trois textures (gauche).

Conditions pratiques

Ces différents critères de similarité impliquent, en définitive, deux conditions quant au zoom infini du type "Dynamic Canvas" : toute image I devant ressembler à T en termes de couleurs, de fréquence et de distribution du « motif ».

Il faut, tout d'abord, que l'algorithme du zoom infini limite le mélange des octaves, sans quoi de nouvelles couleurs et fréquences vont être créées : les histogrammes et spectres divergeant. Cela suppose soit de trouver une « fonction de mélange » minimisant le moyennage des couleurs, soit de limiter le nombre d'octaves considérées simultanément.

Il est ensuite nécessaire de limiter le recouvrement du « motif » des différentes octaves, afin que celui d'origine soit toujours distinguable. Cette condition impose soit l'alignement du motif des différentes octaves – quand un tel alignement peut être trouvé –, soit, à nouveau, de limiter le nombre d'octaves.

3.2.2 Critères dynamiques

Les critères dynamiques doivent permettre de définir la qualité du zoom infini au cours de l'animation. Ils doivent, par conséquent, tenir compte des exigences imposées par la cohérence temporelle tout en maintenant l'illusion du zoom.

Conditions imposées par la cohérence temporelle

La cohérence temporelle impose de limiter au maximum les apparitions brusques et aléatoires de nouveaux éléments, pour éviter les clignotements à l'écran. Cela implique une certaine continuité au cours de l'animation et donc la présence simultanée d'un nombre suffisant d'octaves – pour que l'observateur ne perçoive pas le rafraîchissement du « motif »– mélangées de façon douce.

Conditions liées à l'illusion de zoom

L'animation doit donner l'illusion du zoom : les éléments devant sembler grossir au cours du temps. Cependant, il est souhaité de conserver en permanence une taille et une distribution fréquentielle du « motif » globalement constante à l'écran. Il s'agit du principal paradoxe du zoom infini qui implique l'utilisation d'un cycle fini entre un nombre limité d'octaves.

3.2.3 Bilan

Outre le fait que l'illusion du zoom infini bidimensionnel soit intrinsèquement paradoxale, les critères de qualité statiques et dynamiques que nous avons pu définir, et les contraintes qu'ils imposent, sont eux-mêmes contradictoires.

De plus, ces critères ne sont pas nécessairement les mêmes pour tout type de textures ou de « motif » : les critères de similarité en termes de couleurs, de fréquence ou de distribution n'ayant pas la même importance pour chacun d'eux.

Il existe une telle diversité de textures qu'une solution idéale au problème du zoom infini bidimensionnel s'avère impossible à définir. Il est cependant concevable d'améliorer l'algorithme de "Dynamic Canvas" en fonction de la classe de la texture considérée.

Ainsi, pour résoudre notre problème, il nous faut avant tout trouver le meilleur compromis entre les contraintes statiques et dynamiques précédemment définies, en fonction du type de textures considéré. C'est pourquoi nous proposons dans la section suivante de définir une classification des textures adaptée à notre problème, avant de déterminer ce compromis.

3.3 Classes de textures

L'analyse et la classification de texture est un domaine de recherche vaste dont nous ne ferons pas ici une description détaillée. Nous proposons uniquement une rapide présentation de deux grandes approches d'analyse de textures issues de deux branches distinctes de l'informatique graphique.

Le premier propose, en effet, une modélisation mathématique des textures en utilisant différents outils (géométrie, statistiques, analyse spectrale...), tandis que le second se base sur la perception humaine pour en déduire des mesures caractéristiques.

En s'inspirant de ces modèles, nous proposerons alors une classification simple et intuitive, adaptée au problème du zoom infini bidimensionnel.

3.3.1 Modèles mathématiques

Le but de l'analyse de texture est de formaliser les descriptifs de la texture par des paramètres mathématiques qui serviraient à l'identifier. Une multitude de méthodes, de variantes et de combinaisons de méthodes ont déjà été proposées dans la littérature et éprouvées en pratique [88].

Une classification, couramment utilisée en analyse et synthèse de textures, en a été déduite. Après avoir introduit succinctement les grandes classes de méthodes d'analyse, nous présenterons cette classification.

Méthodes d'analyse

Parmi toutes les méthodes d'analyse mathématique de textures, trois grandes approches sont distinguables : les méthodes structurelles ou géométriques, les méthodes statistiques ou probabilistes et les méthodes spatio-fréquentielles.

- Méthodes structurelles / géométriques

Ces méthodes sont particulièrement bien adaptées aux textures macroscopiques, *i.e.,* présentant un aspect régulier, sous formes de motifs répétitifs, spatialement placés selon une règle précise (peau de lézard, mur de brique, mosaïques (fig. 3.9(a)). . .).

Ces méthodes procèdent généralement en deux étapes. Elles identifient tout d'abord les structures macroscopiques constitutives de la texture – par reconnaissance de forme, par exemple –, puis définissent les règles de placement de ces éléments.

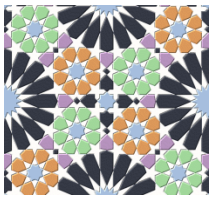
- Méthodes statistiques / probabilistes

Selon les méthodes statistiques, la texture est considérée comme la réalisation d'un processus stochastique stationnaire[2]. Les approches probabilistes cherchent alors à caractériser l'aspect anarchique et homogène de ces textures, des paramètres statistiques (matrice de coocurrences, fonction d'autocorrélation, modèle de Markov. . .) étant estimés pour chaque pixel de l'image.

Ces approches sont donc particulièrement adaptées aux textures présentant des structures « microscopiques » distribuées de manière aléatoire (sable (fig. 3.9(b)), granite, herbe. . .).

- Méthodes spatio-fréquentielles

Les représentations spatio-fréquentielles préservent à la fois les informations globales et locales, elles sont donc bien adaptées aux signaux quasi périodiques. De nombreuses textures sont des signaux quasi périodiques qui ont une énergie fréquentielle localisée (fig. 3.9(c)). Ces méthodes permettent alors de caractériser ce type de textures à différentes échelles, par analyse de leur décomposition dans le domaine de Fourier, et surtout, sur une base de Gabor ou d'ondelettes.



(a) Texture de mosaïque.

(b) Texture de sable.

(c) Texture de tissage.

FIG. 3.9: Exemples de textures pour les trois méthodes d'analyse mathématique.

Classification

Les textures sont traditionnellement séparées selon cinq catégories (fig. 3.10) :

- **régulières**, présentant un motif géométrique précis, répété selon des règles de placement bien définies,
- **quasi régulières**, possédant une sorte de « motif » répété à une légère déformation près,
- **irrégulières**, contenant des éléments de « motif » clairement distinguables, mais non-identiques, avec une répétition non-régulière mais cohérente,
- **quasi stochastiques**, présentant des formes caractéristiques mais distribuées de manière aléatoire,
- **stochastiques**, constituées de grains répartis de façon totalement aléatoire.

[2]Ensemble de variables aléatoires X_t dont la loi ne dépend pas de t.

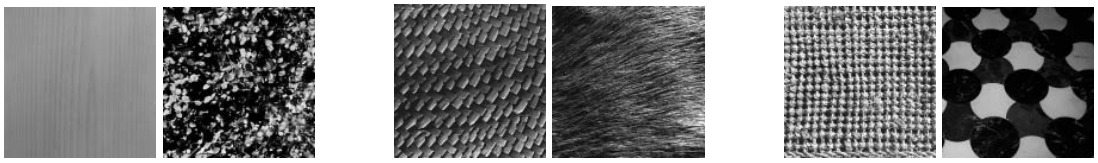
FIG. 3.10: Exemples de textures de la *PSU Near-Regular Texture Database* (51) pour les cinq classes du modèle mathématique.

3.3.2 Modèles perceptuels

Partant du constat que nous sommes bien plus performants pour reconnaître une texture que pour la caractériser – comme l'atteste la profusion de définitions de cette notion en vision par ordinateur –, plusieurs études ont été menées pour établir une classification perceptuelle des textures.

La première étape de ces approches est de définir les indices peceptuels qualifiant une texture. Les travaux fondateurs de Tamura *et al.* [86] décrivent et proposent la mesure de six indices perceptuels, validés par une étude sur 48 utilisateurs. Ils caractérisent ainsi une texture par son degré de :

- rugosité,
- contraste (fig. 3.11(a)),
- directionnalité (fig. 3.11(b)),
- « linéarité » (quantité de lignes droites dans les contours de la texture),
- régularité,
- granularité (fig. 3.11(c)).



(a) Contraste (b) Directionalité (c) Granularité

FIG. 3.11: Exemples de textures de la base de donnée *Vis Tex* (1) pour un faible (gauche) et un fort (droite) taux d'un indice perceptuel.

Rao et Lohse [78] restreignent ces indices à trois critères discriminants principaux – la directionnalité, la périodicité et la complexité – qu'ils jugent suffisamment orthogonaux pour construire une base de l'espace des textures. À travers une nouvelle étude utilisateur [79], ils définissent huit catégories de textures basées sur ces critères :

- « granuleuses », possédant une forme moyenne de base (un grain) distribuée de façon aléatoire,
- « types marbre », aléatoires, non-répétitives et non-granuleuses,
- « en dentelle », non-aléatoires, non-répétitives et sans direction privilégiée,
- « alétoires homogènes », avec des détails fins mais pas de structure,
- « aléatoires répétitives », mais sans régularité,
- « localement directionnelles »,

- « répétitives directionnelles »,
- « répétitives non-directionnelles ».

Les indices perceptuels précédents, et les classifications qui en découlent, ont été utilisés avec succès pour améliorer la recherche de textures dans une base de données [5, 59], mais restent complexes à mesurer.

3.3.3 Classification adaptée à notre problème

Les deux types d'approches décrites précédemment, et les classifications qui en ont été déduites, nous ont servi d'inspiration pour construire notre propre classification. Même si cela n'est pas l'objet de notre étude, il est à noter qu'il semble tout à fait envisageable d'utiliser des algorithmes tirés de ces approches pour réaliser automatiquement la classification d'un ensemble de textures selon notre hiérarchie.

Afin d'obtenir une classification plus simple que les précédentes, nous proposons de regrouper certaines catégorie pour finalement diviser les textures en deux classes principales : celle des textures structurées et celle des textures non-structurées.

La classe des textures structurées regroupe les textures régulières, quasi régulières et irrégulières de la classification mathématique, qui ont toutes en commun la caractéristique de partager un « motif » clairement identifiable. Ce « motif » peut être plus ou moins périodique, sa granularité et sa directionnalité plus ou moins grande, au sens des indices visuels du modèle perceptuel.

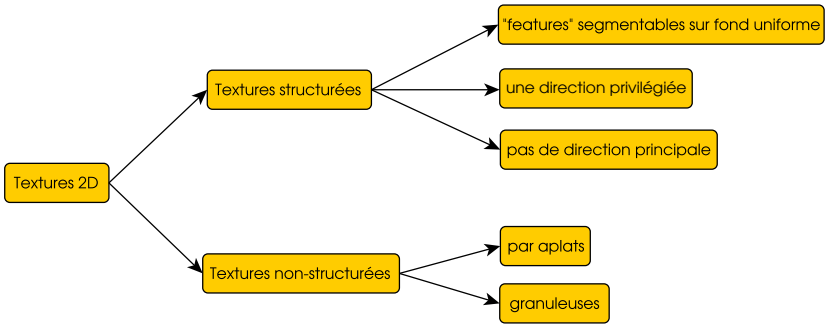
C'est pourquoi nous avons subdivisé la classe des textures structurées selon deux critères : d'une part, la possibilité d'en extraire des marques ("features") clairement segmentables – degré de granularité – sur un fond uniforme (fig. 3.12(a)) et, d'autre part, la présence (fig. 3.12(b)) ou non (fig. 3.12(c)) d'une direction privilégiée – degré de directionnalité.

(a) "Features" segmentables (b) Une direction privilégiée (c) Sans direction privilégiée (d) Granuleuse (e) Par aplats

FIG. 3.12: Exemples de textures illustrant notre classification.

La classe des textures non-structurées englobe les textures quasi stochastiques et stochastiques du modèle mathématique. À nouveau, en fonction de la granularité (la notion de periodicité et de directionnalité ayant peu de sens pour ces textures), deux catégories peuvent être dégagées. Les textures rugueuses avec une fine granularité (fig. 3.12(d)) – les « granuleuses » et « aléatoires homogène » du modèle perceptuel – et celles par aplats ayant une granularité plus forte (fig. 3.12(e)) – la classe perceptuelle des « types marbre ».

La hiérarchie suivante résume notre classification :



3.4 Étude du zoom infini sur le modèle "Dynamic Canvas"

Maintenant que nous avons été capable de définir précisemment une classification des textures adaptée à notre problème, nous proposons d'étudier plus en détails la méthode de zoom infini du type "Dynamic Canvas" en fonction des classes de texture.

"Dynamic Canvas", tel qu'il est proposé par Cunzi *et al.*, comporte un certain nombre de paramètres, assez arbitrairement fixés par les auteurs. Afin de déterminer leur meilleur valeur pour chaque classe de texture, nous avons passé en revue ces leviers de contrôle et comparé les résultats obtenus en les faisant varier.

Nous avons pu mettre à jour cinq paramètres principaux sur lesquels influer : la nature, le nombre et la taille relative des octaves utilisées, leur vitesse de grossissement et la fonction de mélange de ces octaves.

3.4.1 Nature et taille relative des octaves

Dans un cas idéal, la texture sur laquelle on souhaite zoomer représenterait un motif fractal, parfaitement autosimilaire. Une seule octave serait alors nécessaire et, après un certain facteur de zoom, l'image originale serait naturellement retrouvée (fig. 3.13).

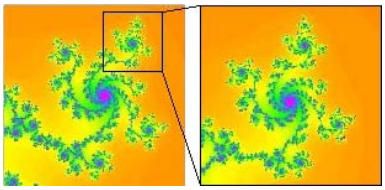


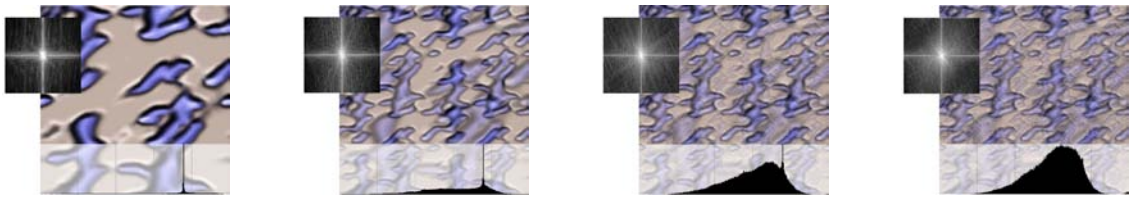
FIG. 3.13: Zoom sur un motif fractal.

Dans un cas général, le « motif » – qui ne peut être véritablement défini que dans le cas des textures structurées – de la texture n'a aucune raison d'être autosimilaire. L'idée centrale de "Dynamic Canvas" est de mélanger la même texture à différentes échelles pour créer artificiellement l'autosimilarité. Afin d'obtenir un cycle infini continu, il est nécessaire que le facteur d'échelle entre les octaves soit de deux, même si un facteur plus important aurait augmenté la similarité du mélange avec la texture d'origine.

La nature et la taille des octaves choisies dans l'approche initiale semblent donc tout à fait justifiées, aucune alternative n'étant capable de produire le résultat escompté, quelque soit le type de texture.

3.4.2 Nombre d'octaves

Les Figures 3.14 et 3.15 montrent le résultat du mélange avec les paramètres précédents pour n octaves, ainsi que leur histogramme moyen pour les trois canaux et leur spectre. Force est de constater qu'il y a d'autant moins de moyennage des couleurs (étalement de l'histogramme) et d'apparition de fréquences, et donc le mélange est d'autant plus similaire à la texture originale, qu'il y a peu d'octaves.

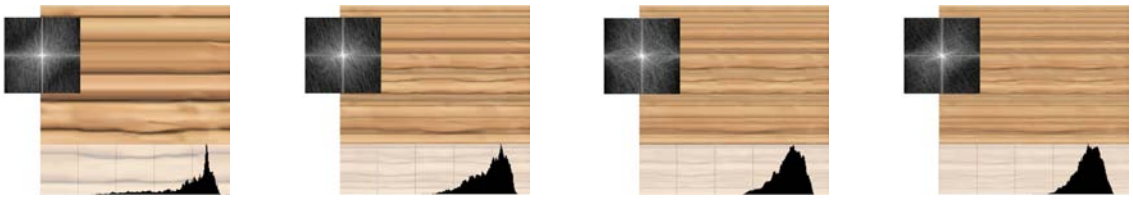


(a) Une octave (b) Deux octaves (c) Trois octaves (d) Quatre octaves

FIG. 3.14: Résultat du mélange pour n octaves avec une texture à "features" segmentables ainsi que leur histogramme moyen sur les trois canaux et leur spectre.

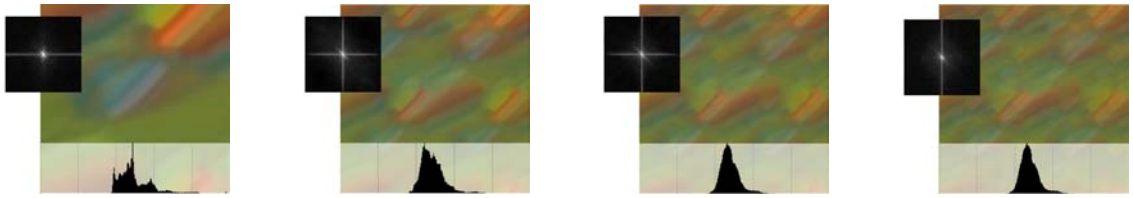
On constate néanmoins que, visuellement et du point de vue de l'histogramme, le mélange dégrade beaucoup moins la similarité avec la texture d'origine dans le cas des textures présentant une direction privilégiée (fig. 3.15) que dans celui de "features" segmentables, car le « motif » est naturellement aligné à toutes les échelles.

La même remarque est valable pour les textures non-structurées, en particulier celles par aplats (fig. 3.16), l'apparition (modérée) de nouvelles fréquences ou couleurs n'étant pas visuellement très perturbantes.



(a) Une octave (b) Deux octaves (c) Trois octaves (d) Quatre octaves

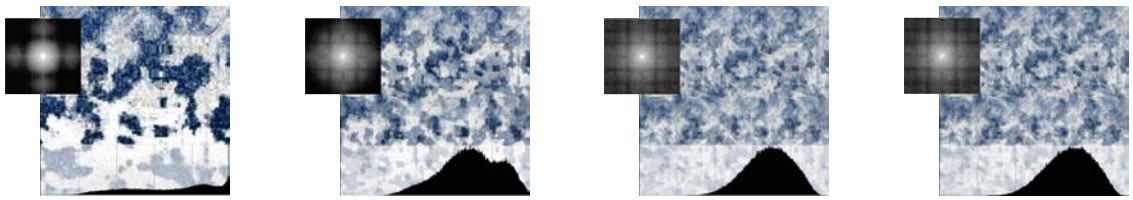
FIG. 3.15: Résultat du mélange pour *n* octaves avec une texture possédant une direction privilégiée ainsi que leur histogramme moyen sur les trois canaux et leur spectre.



(a) Une octave (b) Deux octaves (c) Trois octaves (d) Quatre octaves

FIG. 3.16: Résultat du mélange pour *n* octaves avec une texture non-structurée par aplats ainsi que leur histogramme moyen sur les trois canaux et leur spectre.

Dans le cas de textures granuleuses (fig. 3.17), l'étalement de l'histogramme et du spectre est plus important, mais sans avoir un impact visuel trop important. En effet, dans la mesure où la textures d'origine possède déjà une gamme de fréquences assez large, la reconnaissance de la texture d'origine n'est pas tellement remise en cause par le mélange.



(a) Une octave (b) Deux octaves (c) Trois octaves (d) Quatre octaves

FIG. 3.17: Résultat du mélange pour *n* octaves avec une texture non-structurée granuleuse ainsi que leur histogramme moyen sur les trois canaux et leur spectre.

Cependant, pour garantir l'impression de continuité temporelle au cours du zoom, il est nécessaire d'utiliser au moins trois ou quatre octaves pour que l'apparition des octaves inférieures soit douce. Il n'est donc pas vraiment possible de restreindre le nombre d'octaves simultanées pour augmenter la similarité avec la texture d'origine – ce qui pourrait pourtant être particulièrement utile dans les cas des textures structurées sans direction privilégiée.

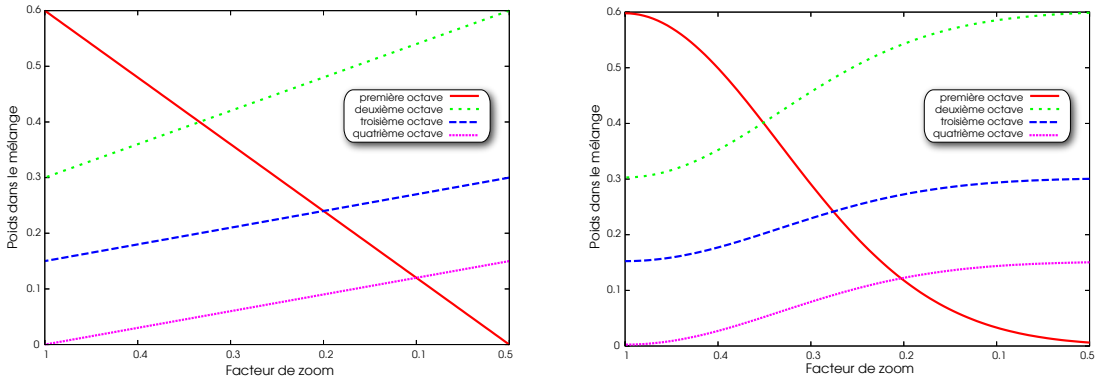
3.4.3 Vitesse de grossissement et fonction de mélange

Cunzi *et al.* ont montré que la vitesse de grossissement doit être logarithmique avec la profondeur (le facteur de zoom étant exponentiel) pour que la taille des octaves à l'écran croisse de façon linéaire. Il est donc impossible de modifier ce paramètre si l'on souhaite un zoom infini continu.

La fonction de mélange, en lien avec l'enveloppe spectrale, détermine la façon dont les octaves sont combinées. Il s'agit donc d'un levier important du zoom infini que nous nous proposons de considérer maintenant.

La fonction de mélange proposée dans "Dynamic Canvas" est un fondu linéaire entre les octaves (fig. 3.18(a)) : fonction qui a montré ses limites dans les exemples précédents. L'enveloppe fréquentielle contrôlant l'apparition des octaves était une gaussienne dans l'approche unidimensionnelle de Shepard [82]. Cependant, déjà en dimension un, Purwins [75] a montré que la forme exacte de l'enveloppe

n'avait pas une influence significative sur l'illusion finale.



(a) Pondération linéaire.

(b) Pondération gaussienne.

(c) Enveloppe linéaire.

(d) Enveloppe gaussienne.

FIG. 3.18: Résultat du mélange de 4 octaves pour deux types d'enveloppe fréquentielle : linéaire et gaussienne.

La figure 3.18 confirme ce résultat dans le cas bidimensionnel. Nous avons pondéré le mélange de quatre octaves par une fonction gaussienne (fig. 3.18(b)), de la forme :

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}}, \quad \text{avec } \sigma \text{ l'écart-type.}$$

Aussi bien visuellement qu'en terme d'histogramme ou de spectre, il n'y a pas une franche différence entre un mélange obtenu avec une enveloppe linéaire (fig. 3.18(c)) et un enveloppe gaussienne (fig. 3.18(d)). Quelle que soit l'enveloppe spectrale, un étalement de l'histogramme lié au moyennage des couleurs est en effet observé.

De même, bien que la continuité du raccord de la fonction de mélange gaussienne soit supérieure à celle du mélange linéaire, aucun gain en terme de continuité au cours du zoom n'est visuellement observable.

3.4.4 Bilan

À l'issue de cette étude détaillée du zoom infini bidimensionnel de type "Dynamic Canvas", nous pouvons tirer les conclusions suivantes. Si, cette approche par mélange d'octaves se prête très bien aux textures non-structurées ou structurées possédant une direction principale, elle induit d'importants artefacts pour les classes de textures ne présentant pas un alignement multi-échelle intrinsèque de leur « motif ».

Plus précisément, nous avons pu constater que, dans le cas des textures non-structurées ou structurées avec une direction privilégiée (en vert sur la figure 3.19), cet algorithme est assez approprié, les deux seuls réglage restant étant : le nombre exact d'octaves mélangées (trois ou quatre) et l'enveloppe spectrale (linéaire ou gaussienne) en fonction de la continuité recherchée lors de l'animation.

Par contre, pour des textures structurées possédant des “features” segmentables ou pas de direction privilégiée (en orange sur la figure 3.19) – dont le « motif » est fortement distinguable et donc fortement perturbé par le fondu –, le mélange basique engendre deux problèmes : non seulement le « motif » s’auto-recouvre, mais son contraste est également significativement affaibli. Pour essayer de corriger ces artefacts, nous proposons, pour ces deux classes de textures, de nouvelles solutions de mélange des octaves.

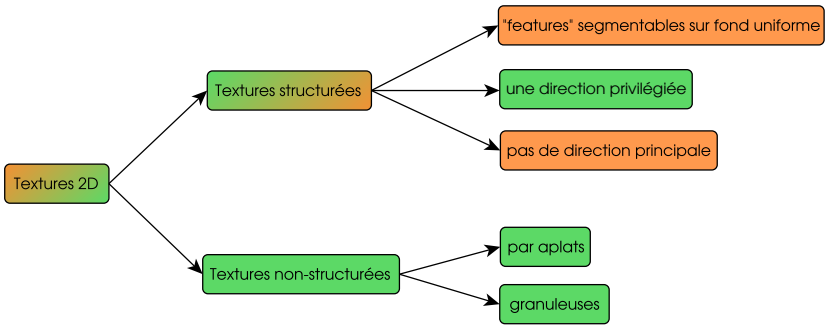


FIG. 3.19: Qualité du mélange du type “Dynamic Canvas” en fonction de notre classification des textures : d’assez bonne pour les classes en vert à mauvaise pour celles en orange

Chapitre 4

Nouvelles solutions au problème du zoom infini

Dans ce chapitre nous proposons de nouvelles solutions pour zoomer infiniment sur une texture 2D structurée, à “features” segmentables ou sans direction privilégiée. En se basant sur les conclusions de l'étude détaillée de “Dynamic Canvas”, effectuée au chapitre précédent, nous présenterons les approches que nous avons jugées à même de produire un compromis visuellement satisfaisant.

4.1 Utilisation d'informations complémentaires

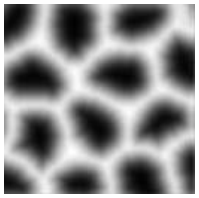
Partant du constat qu'un mélange naïf de n octaves auto-similaires à la manière de “Dynamic Canvas” n'est pas adapté aux textures structurées, à “features” segmentables ou sans direction privilégiée, il nous a semble qu'une piste intéressante serait d'intégrer au mélange des informations complémentaires, extraites spécifiquement de chacune des textures considérées.

Deux informations nous semblent potentiellement utiles pour améliorer l'algorithme du zoom infini : celle de couleur et celle de « motif » de la texture originale. Ces données pourraient être utilisées aussi bien durant le mélange qu'en pré ou post-traitement.

L'information de couleur est facilement accessible – nous l'avons déjà utilisée tout au long du chapitre 3 comme critère de qualité – sous la forme de l'histogramme moyen ou d'un histogramme par canal de couleur.

L'information de « motif » est plus difficile à extraire de la texture, même si notre objectif n'est pas d'en obtenir une segmentation stricte. Wu et Yu [99], puis Lefebvre et Hoppe [54], ont cependant proposé une méthode de « segmentation douce » d'une texture sous la forme d'une *feature map*. La construction de cette carte est obtenue par une suite de traitements d'image afin de rehausser puis détecter les contours du « motif », avant de calculer la distance signée entre ces structures (fig. 4.1).

(a) Texture d'œufs.



(b) *Feature map.*

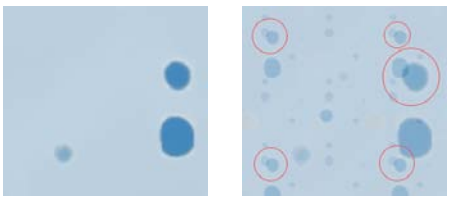
FIG. 4.1: Texture d'œufs et sa *feature map*.

En utilisant ces données supplémentaires, nous avons pu proposer trois extensions à l'algorithme de “Dynamic Canvas” selon différents axes (potentiellement combinables) : l'alignement du « motif » par déformation des octaves – dans le cas des textures segmentables sur fond uniforme –, l'amélioration du contraste du mélange – en particulier en tenant compte de la *feature map* – et l'optimisation de la texture exemple pour maximiser la similitude du mélange avec l'original.

4.2 Alignement du « motif » par déformation des octaves

Il est apparu, au cours de nos différentes tentatives d'amélioration du zoom infini type "Dynamic Canvas", que le principal désagrément visuel lié à cette méthode est la superposition du « motif », dans le cas des textures structurées à "features" segmentables ou ne possédant pas de direction principale.

Dans le cas des textures structurées ne possédant pas de direction principale, la question paraît insoluble sans privilégier arbitrairement l'une des directions, pour autant que l'on soit capable de les extraire.



(a) Texture originale. (b) Mélange.

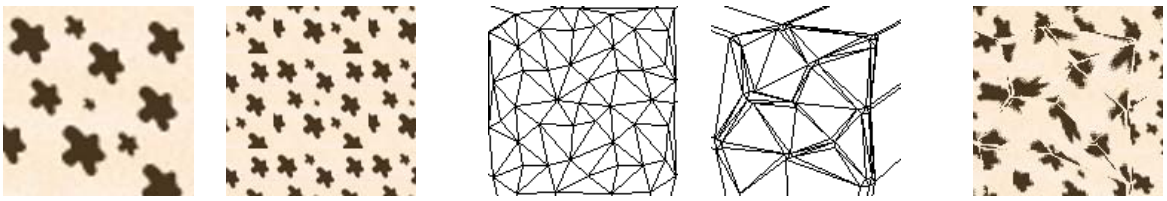
FIG. 4.2: Intersections partielles après le mélange de quatre octaves.

Par conséquent nous nous restreindrons dans la suite au cas des textures possédant des "features" segmentables sur un fond uniforme. Pour cette classe de textures, nous nous somme fixés pour objectif de minimiser les intersections partielles de "features" (fig. 4.2). En effet, ces superpositions ne sont généralement pas présententes dans le « motif » original et le corrompent durant le mélange.

4.2.1 Champ de déformation triangulaire à la manière de Matusik *et al.* (62)

Un problème proche a déjà été traité par Matusik *et al.* [62], dans le cas de textures possédant à l'origine des structures similaires. Notre cas est cependant plus complexe que le leur dans la mesure où les structures de nos différentes octaves – bien qu'identiques à un facteur d'échelle près – peuvent ne présenter initialement aucun alignement.

En nous inspirant de Matusik *et al.*, nous avons mis au point une méthode de déformation basée sur une triangulation. L'idée générale est de déformer la texture de l'octave inférieure Ω_2 de telle sorte que ses "features" soient alignées, au début du zoom, avec celles de l'octave supérieure Ω_1. Durant le zoom, les "features" retrouvent continûment lors position initiale pour permettre le cycle des octaves.



(a) Octaves Ω_1 et Ω_2 initiales. (b) Triangulation de Ω_2 avant (gauche) et après déformation maximale (droite). (c) Ω_2 déformée à l'état initial.

FIG. 4.3: Principales étapes de notre méthode de déformation par triangulation.

Le centre des "features" peut être simplement extrait de la *feature map* en cherchant ses extremums locaux (minimums si les "features" sont codées en noir) par des opérations morphomathématiques (en utilisant la librairie IPL98 [16]). La triangulation de Delaunay des centres de l'octave Ω_2 est alors calculée selon l'algorithme incrémental de Lischinski [57]. Le plus proche voisin de chacun de ces points dans l'octave supérieure Ω_1 (trouvé par la librairie ANN [64]) nous permet de définir le déplacement maximal, conduisant à la triangulation déformée. La figure 4.3 résume graphiquement ce processus.

Au cours du zoom infini, la triangulation retrouve progressivement sa position initiale par interpolation linéaire de ses sommets. L'octave inférieure déformée $\hat{\Omega}_2$ est construite par morphage barycentrique de Ω_2 selon cette triangulation.

Cette méthode permet de produire une déformation importante de l'octave inférieure Ω_2 dans la direction des "features" de l'octave supérieure Ω_1 (fig. 4.3(c)) comme nous le souhaitions. En outre, elle produit un retour continu de l'octave Ω_2 à son état initial, le morphage barycentrique étant lui même continu.

Cependant, comme l'illustre la figure 4.4 sur une séquence de zoom, le premier problème de cette approche est la triangulation qui est très visible dans le mélange final. Pour pallier ce problème, il serait possible de subdiviser le maillage et d'interpoler le déplacement aux points de subdivision. Néanmoins, l'effet de contraction très géométrique de l'octave inférieure Ω_2 vers les extremums de l'octave supérieure Ω_1 ne pourra pas être totalement évité.

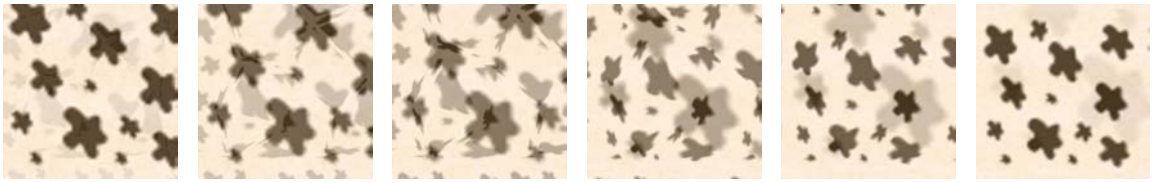


FIG. 4.4: Séquence de zoom obtenue avec notre méthode de déformation par triangulation.

En outre, on peut constater que la déformation ne correspond pas exactement à nos attentes. En effet, les "features" de Ω_2 sont étirées et non déplacées sous celles de Ω_1 pour éviter les intersections partielles. Au regard de ces problèmes, nous avons développé une nouvelle approche, plus douce car ne reposant pas sur une structure géométrique, pour permettre cet alignement.

4.2.2 Déformation par advection

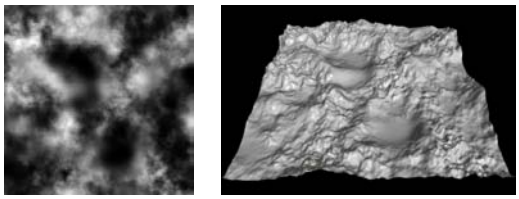
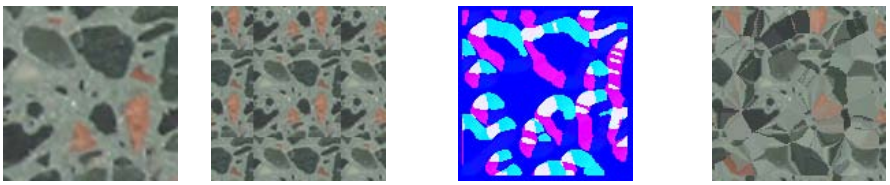


FIG. 4.5: Carte de hauteur et le terrain associé.

Nous proposons une approche pixellique de déformation en s'inspirant des techniques d'advection de texture (pour de plus amples détails concernant cette méthode cf. section 2.2.3). L'idée est de considérer la *feature map* de l'octave supérieure Ω_1 comme une carte de hauteur – image utilisée pour stocker une information d'élévation : altitude d'un terrain, par exemple (fig. 4.5) – les "features" de l'octave inférieure Ω_1 devant se déplacer vers les extremums de cette carte.

La direction de déplacement peut directement être calculée depuis la *feature map* d'Ω_1 en prenant sa dérivée bidimensionnelle selon les axes x et y – *i.e.,* le gradient de cette image (fig. 4.6(b)) – qui indique le sens de la plus grande pente. La déformation de l'octave Ω_2 est alors obtenue par *inverse warping* : en déterminant pour tout point (x',y') de l'image défomée, le pixel $(x,y) = W^{-1}(x',y')$ dans l'image d'origine.



(a) Octaves Ω_1 et Ω_2 initiales. (b) Gradient de la *feature map* d'Ω_1. (c) Ω_2 déformée à l'état initial.

FIG. 4.6: Principales étapes de notre méthode de déformation par advection.

La figure 4.7 montre une séquence de zoom obtenue par cette méthode. La déformation au cours du temps est beaucoup plus continue que l'approche précédente par triangulation et les "features" de l'octave inférieure se déforment vraiment pour s'aligner sur celles de l'octave supérieure.

Deux limitations sont cependant à noter. La première concerne l'impression de flou qui reste encore très grande avec cette méthode. En effet, si les octaves sont correctement alignées à l'état initial du zoom infini en limitant les intersection partielles, celles-ci réapparaissent au cours de l'animation, Ω_2 devant retrouver sa position non-déformée pour permettre le cycle des octaves.

FIG. 4.7: Séquence de zoom obtenue avec notre méthode de déformation par advection.

La seconde limitation est commune aux deux approches par déformation et concerne l'ajout des octaves suivantes Ω_i pour $i > 2$. Il n'est pas évident de déterminer qu'elle serait la « bonne » position de ces octaves, non seulement à l'état initial du zoom, mais aussi tout au long de la déformation, ce qui rend ce problème difficile. C'est pourquoi, de nombreuses pistes restent encore à explorer autour de ces approches et constituent d'intéressants travaux futurs.

4.3 Optimisation de la texture originale pour le mélange

Étant donné que le « motif » des textures utilisées peut être quelconque, nous avons fixé comme postulat de base que le mélange devait nécessairement s'effectuer, comme dans "Dynamic Canvas", sur n copies – à des échelles différentes – de la texture originale T. Rien ne nous empêche cependant de modifier cette texture de telle sorte que le mélange de ses n octaves soit le plus similaire possible à l'original.

Il s'agit en fait d'un problème d'optimisation : nous cherchons la texture $\hat{T}$ minimisant $||T - \Sigma_{\hat{T}}||$ avec :

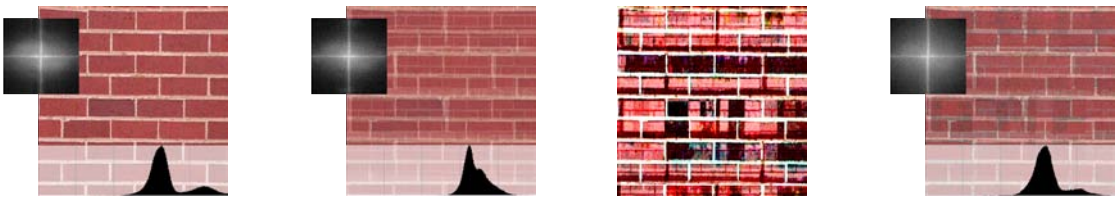
$$\Sigma_{\hat{T}}(i,j) = \alpha_1 \hat{T}(i/2, j/2) + \alpha_2 \hat{T}(i,j) + \alpha_3 \hat{T}(2i,2j) + \alpha_4 \hat{T}(4i,4j) \quad \forall (i,j) \tag{4.1}$$

les α_k correspondant à la pondération du mélange. Pour une pondération fixée – *i.e.,* une image fixe du zoom infini – déterminer $\hat{T}$ revient à résoudre un système linéaire de la forme $A\hat{t} = t$, t et $\hat{t}$ étant les versions vecteurs des images T et $\hat{T}$ et la matrice carrée A contenant la pondération telle que $A\hat{t} = \sigma_{\hat{T}}$ ($\sigma_{\hat{T}}$ étant la version vecteur de $\Sigma_{\hat{T}}$).

Nous proposons de résoudre ce système par la méthode itérative de Gauss-Seidel, l'équation 4.1 pouvant se réécrire sous la forme :

$$\hat{T}(i,j) = \frac{T(i,j) - \alpha_1 \hat{T}(i/2, j/2) - \alpha_3 \hat{T}(2i,2j) - \alpha_4 \hat{T}(4i,4j)}{\alpha_2} \quad \forall (i,j) \tag{4.2}$$

en fixant $\Sigma_{\hat{T}} = T$, cas idéal si la similitude était totale.



(a) Texture originale. (b) Mélange original. (c) Texture optimisée. (d) Mélange optimisé.

FIG. 4.8: Résultat du mélange avec quatre octaves pour une texture de brique et sa version optimisée.

La figure 4.8 montre le résultat obtenu avec cette méthode pour une texture structurée possédant un certain alignement. Il est à noter que l'histogramme moyen du mélange optimisé (fig. 4.8(d)) est bien plus proche de celui de la texture originale que ne l'est celui du mélange par fondu.

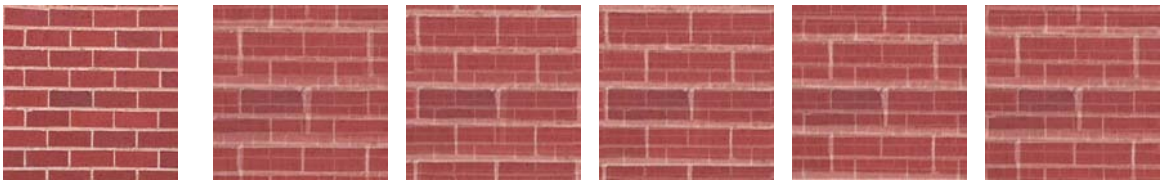
Spectralement et visuellement, la comparaison est plus difficile. En effet, comme on peut le voir sur la figure 4.8(c), l'optimisation rehausse le contraste du « motif » de l'octave la plus visible pour la pondération courante, tout en essayant de compenser celui des autres octaves. Or, cette balance n'étant pas parfaite, de nouveaux artefacts (traces noires) sont créés. Néanmoins, l'utilisation d'une méthode

d'optimisation plus douce devrait permettre une réduction de cet effet.

La principale difficulté est désormais de passer de l'optimisation d'une image fixe – pour une pondération donnée – à une optimisation dynamique sur la totalité du zoom. En effet, utiliser la texture optimisée par la méthode précédente au cours du zoom infini produit un résultat de piètre qualité (fig. 4.9(d)). Il n'est cependant pas possible d'optimiser une texture pour chaque pondération : la continuité du zoom, et donc la cohérence temporelle, ne serait alors plus garantie.

Une fois encore, il s'agit de trouver le meilleur compromis entre similarité avec l'original et continuité temporelle. Nous proposons pour cela de faire varier la pondération du mélange au cours des itérations de l'algorithme d'optimisation de la même façon qu'elle évolue au cours du zoom. Ainsi, la texture optimisée devrait constituer un compromis acceptable à tout instant de l'animation.

La séquence de zoom de la figure 4.9(f) montre la très nette amélioration de la qualité du mélange obtenu avec cette optimisation dynamique. Il est important de souligner que le « motif » des octaves inférieures est toujours présent mais est compensé dynamiquement pour renforcer en permanence la similarité avec l'original, quelle que soit l'octave possédant la plus forte pondération dans le mélange.



(a) Texture originale. (b) Zoom infini sans optimisation.

(c) Opt. statique. (d) Zoom infini après optimisation statique.

(e) Opt. dynamique. (f) Zoom infini après optimisation dynamique.

FIG. 4.9: Séquences de zoom infini obtenues sans optimisation, par optimisation statique d'une image fixe, puis dynamique sur l'ensemble de l'animation.

Cette approche n'est cependant pas valide pour toutes les textures. Elle ne corrige en effet pas les chevauchements partiels, dans le cas des textures à "features" segmentables (fig. 4.10(a)). En outre, elle peut introduire de nouvelles couleurs en voulant surcompenser le mélange (fig. 4.10(b)). Enfin, elle n'apporte pas un gain significatif aux textures ne présentant aucun alignement (fig. 4.10(b)).

(a) Texture à "features" segmentables. (b) Texture structurée sans direction privilégiée.

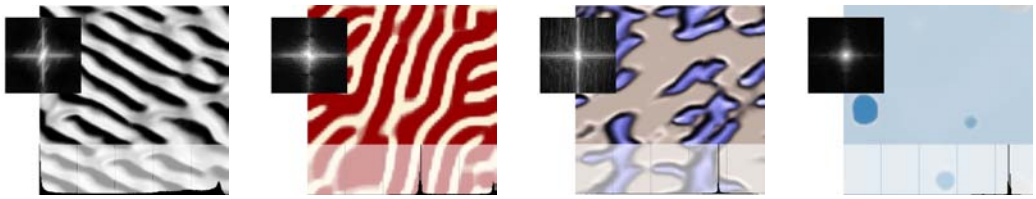
FIG. 4.10: Exemple de cas problématiques pour notre approche par optimisation (original à droite et mélange à gauche).

4.4 Amélioration du contraste du mélange

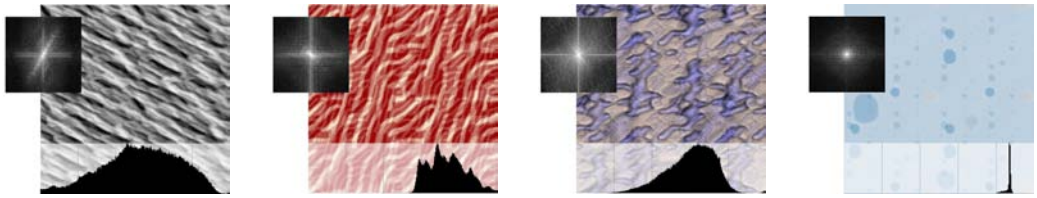
La dernière piste que nous avons explorée est l'amélioration du contraste du « motif » de la texture, non pas par pré-traitement de l'original comme dans l'approche précédente par optimisation, mais durant le mélange. Nous proposons ainsi deux nouvelles fonctions de mélange pour remplacer le fondu linéaire de "Dynamic Canvas".

4.4.1 Mélange par seuillage

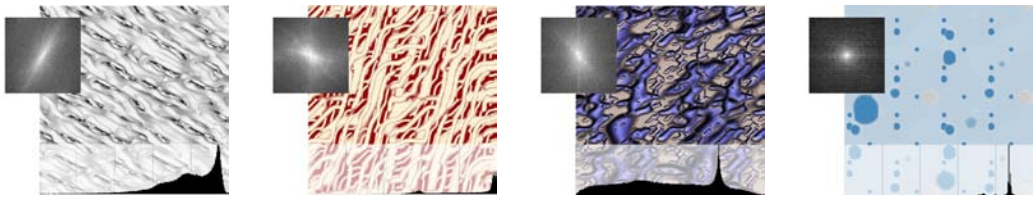
Dans le cas des textures structurées sans direction privilégiée – les plus dégradées par l'algorithme actuel – qui sont quasi binaires (avec deux couleurs dominantes), nous proposons d'utiliser un mélange par seuillage. Sachant que le blanc correspond au triplet RGB $(255, 255, 255)$ et le noir au triplet $(0, 0, 0)$, il consiste à prendre les minimums (pour les textures au « motif » foncé sur fond clair) ou les maximums (pour les textures au « motif » clair sur fond foncé) des n octaves du mélange (fig. 4.11(c)).



(a) Textures originales.



(b) Mélange par fondu.



(c) Mélange par seuillage.

FIG. 4.11: Résultats obtenus, pour quatre textures structurées, par les deux méthodes de mélange : le fondu et le seuillage.

Par comparaison avec la figure 4.11(b), on peut observer un gain certain de contraste, les histogrammes des textures seuillées étant moins étalés que ceux obtenus par fondu. Ce résultat est tout à fait cohérent, dans la mesure où le moyennage des couleurs est évité. Cependant, exception faite de la texture de bulles (à droite) possédant des "features" très clairement segmentables, le contraste est gagné aux dépends de l'une des deux couleurs, l'autre disparaissant presque totalement.

En outre, force est de reconnaître que le seuillage dégrade encore plus la similarité du mélange avec l'original, particulièrement en termes de fréquences. En effet, si le seuillage conserve le « motif » de toutes les octaves, il n'évite pas les superpositions d'une couche sur l'autre, créant ainsi un effet en « couches d'oignon » indésirable.

De plus, dynamiquement, la continuité du zoom est bien inférieure à l'approche par fondu : les octaves extrêmes apparaissant ou disparaissant de façon assez brutale.

Ce mélange par seuillage, s'il apporte un gain en terme de contraste, se comporte de façon trop

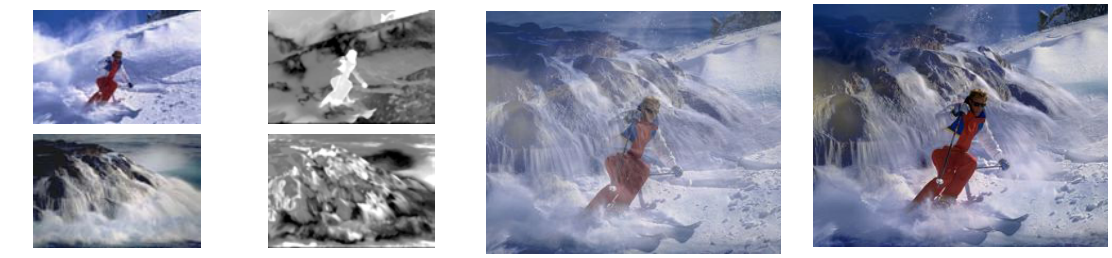
binaire en ne privilégiant qu'une seule couleur du « motif ». Pour que notre solution soit véritablement adaptée à chaque classe de texture, la fonctions de mélange devrait tenir compte du « motif » de celle-ci : c'est ce que nous proposons avec notre mélange « intelligent ».

4.4.2 Mélange « intelligent »

L'objectif de ce mélange est de rehausser ponctuellement le contraste du mélange, et en particulier celui de son « motif », pour qu'il se rapproche, en terme d'histogramme, de la texture originale tout en faisant ressortir les structures remarquables. Il s'agit donc d'obtenir, en lieu et place d'une fonction de mélange globale, une fonction s'adaptant localement au « motif » de la texture, ce qui suppose de disposer de sa *feature map*.

Ce problème est proche de celui de la composition de deux images [10], utilisée, par exemple, au cinéma pour créer des effets spéciaux. La composition se divise en deux étapes : la sélection des parties des images à combiner – généralement de façon manuelle, assistée par ordinateur [2, 70] – et le mélange de ces parties. Traditionnellement, le mélange est réalisé par fondu linéaire entre les deux images [72].

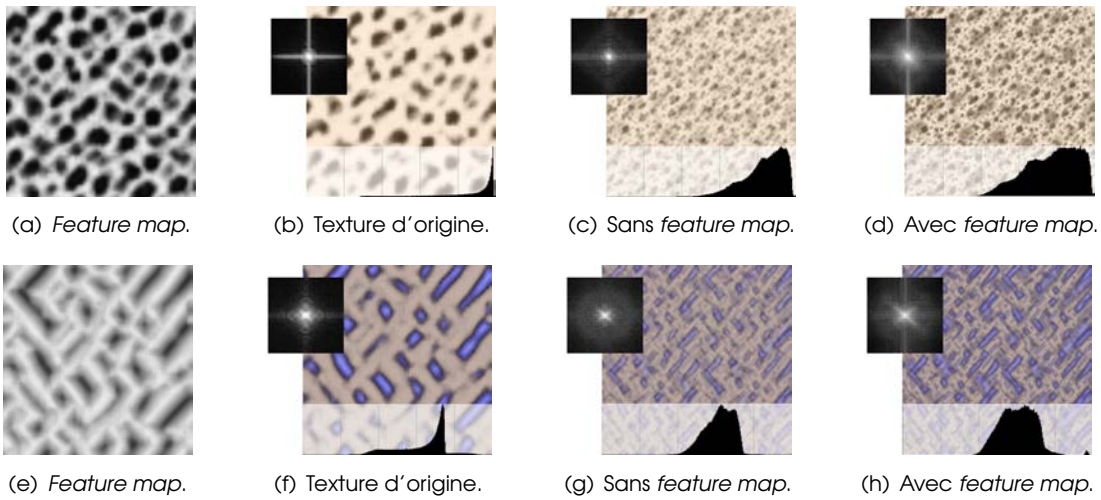
Cependant, une approche récente proposée par Grundland *et al.* [29] introduit des fonctions de mélanges non-linéaires préservant le contraste général, la couleur globale ou les structures remarquables des deux images. Cette dernière fonction prend en entrée une *salience map* – version plus continue de la *feature map* – et produit un mélange tenant compte de cette information, comme l'illustre la figure 4.12.



(a) Images d'origine. (b) *Salience map*. (c) Mélange linéaire. (d) Mélange adaptatif.

FIG. 4.12: Mélange adaptatif selon la méthode de Grundland *et al.* (29).

Nous proposons d'utiliser, de la même façon, la *feature map* de la texture à chaque octave pour rehausser le poids du « motif » (zones sombres dans la *feature map*) par rapport au fond (zones claires) dans la fonction de mélange. La figure 4.13 compare le résultat du mélange avec et sans cette information de « motif » pour deux textures structurées.



(a) *Feature map*. (b) Texture d'origine. (c) Sans *feature map*. (d) Avec *feature map*.

(e) *Feature map*. (f) Texture d'origine. (g) Sans *feature map*. (h) Avec *feature map*.

FIG. 4.13: Comparaison des mélanges obtenus avec et sans l'information de la *feature map*.

On peut observer un gain certain en terme de contraste du « motif », ce qui peut, visuellement, aider à sa reconnaissance. Il faut cependant avouer que du point de vue fréquentiel ou des histogrammes, la similarité avec l'original n'est pas véritablement augmentée.

(a) Textures d'entrée (b) Mélanges sans (gauche) et avec (droite) traitement

FIG. 4.14: Diminution du flou par la méthode de Matusik *et al.* (62)

Une autre approche, ne nécessitant pas explicitement une *feature map* – l'information de « motif » étant extraite par leur algorithme –, a été proposée par Matusik *et al.* [62] pour rehausser les contours de deux textures mélangées linéairement. Pour cela, ils adaptent la méthode de mise en correspondance d'histogrammes introduite en synthèse de texture par De Bonet [32] (plus de détails en section A.1.2). Les résultats qu'ils obtiennent (fig. 4.14) montrent une réelle diminution du flou produit par le mélange.

Nous n'avons cependant pas implémenté cette méthode car, aux regards des résultats obtenus avec la *feature map*, le mélange adaptatif ne nous semble pas suffisant pour apporter une solution pleinement satisfaisante : l'alignement du « motif » créant une gêne visuelle plus importante que celle liée au contraste.

4.5 Discussion

Nous avons montré que les artefacts induits par la fonction de mélange de "Dynamic Canvas" sont de deux natures : l'auto-recouvrement du « motif » et la perte de contraste due au fondu.

Ce dernier problème a pu être résolu en rendant le mélange adaptatif au « motif » de la texture, par l'utilisation d'une *feature map*. La mise en correspondance d'histogramme est une alternative possible, en l'absence de cette information.

En outre, dans le cas des textures à "features" segmentables sur fond uniforme, nous avons montré qu'il est possible de créer cet alignement, par déformation d'une octave vis-à-vis de son octave supérieure. Les méthodes que nous avons proposées démontrent la pertinence de cette approche, même si elles nécessiteraient de plus amples recherches pour en améliorer les résultats, et surtout, s'appliquer à plus de deux octaves.

Nous avons par ailleurs proposé une méthode d'amélioration générale de la qualité du mélange, par optimisation de la texture originale sur la séquence de zoom. Si les résultats obtenus pour certaines textures sont très encourageants, l'utilisation de méthodes d'optimisation plus fines et leur combinaison avec certaines approches précédentes semblent d'autres pistes de recherche intéressantes à poursuivre.

Bien que les solutions que nous avons apportées au problème du zoom infini sur une texture 2D soient encore incomplètes, elles nous semblent déjà suffisantes pour pouvoir envisager le passage à la dimension supérieure.

En effet, la principale limitation de ces méthodes bidimentionnelles est qu'elles imposent l'utilisation d'une seule texture 2D pour tout une scène 3D – comme dans l'approche originale de Cunzi *et al.*. Étendre cette méthodes aux textures volumiques permettrait, à l'inverse, d'assigner une texture 3D infiniment zoomable par objet de la scène et donc de démultiplier la palette des effets de matière possibles, ceux-ci restant temporellement cohérents au cours de l'animation.

Chapitre 5

Extension 3D du zoom infini

L'objectif de ce chapitre est de détailler comment nous avons partiellement étendu les solutions trouvées aux chapitres 3 et 4 au problème du zoom infini bidimensionnel, à la dimension supérieure.

Ce passage à la 3D permet l'utilisation d'une (voire plusieurs) texture(s) par objet – contrairement à l'approche "Dynamic Canvas" qui n'autorise qu'une texture 2D pour toute la scène. De plus, ces textures volumiques suivent parfaitement non seulement le mouvement de la caméra – alors qu'il n'est exact que pour une profondeur fixée par l'utilisateur dans "Dynamic Canvas" –, mais aussi celui des objets 3D – la scène devant être fixe dans la méthode originale. Enfin, ce type de texture à l'avantage de nous affranchir des problèmes de paramétrisation.

Nous avons complètement réalisé cette extension pour la méthode 2D de base – le mélange linéaire de quatre octaves – particulièrement adaptée pour les textures non-structurées ou n'ayant qu'une direction privilégiée (cf. section 5.1), mais proposerons quelques pistes d'extension pour nos autres solutions 2D (cf. section 5.2).

Notre approche étant basée textures, il nous a été nécessaire de synthétiser un certain nombre de textures 3D. De plus amples détails quant aux différentes méthodes de synthèses de textures volumiques disponibles et l'implémentation de l'approche que nous avons choisie, sont disponibles en annexe A.

5.1 Extension à la 3D de la solution par fondu

Après avoir décrit en détails notre adaptation tridimensionnelle de l'approche de zoom infini 2D type "Dynamic Canvas", nous discuterons des avantages et inconvénients de cette approche 3D, en fonction des classes de textures.

5.1.1 Détails de la méthode

Dans le cas 2D de "Dynamic Canvas", la texture est affichée à l'écran et redimensionnée en fonction du déplacement du visiteur virtuel, pour simuler le zoom. À l'inverse, dans le cas tridimensionnel, la texture volumique est directement accrochée à un objet 3D et le redimensionnement est obtenu automatiquement par projection perspective de cet l'objet. Il faut cependant déterminer le facteur de zoom devant être appliqué à la texture volumique pour que la taille à l'écran soit constante.

Pour une texture 3D, le facteur de zoom correspond à la distance de l'objet à la caméra $distC$. Plus l'objet est près de la caméra, plus la fréquence de la texture doit être faible. L'évolution de la fréquence n'est cependant pas linéaire avec le facteur de zoom mais exponentielle, comme l'ont montré Cunzi *et al.* [14].

La partie entière du logarithme en base 2 de la distance caméra-objet $ld = \lfloor \log 2(distC) \rfloor$ indique donc le nombre de fois que la fréquence de la texture doit être doublée pour obtenir une taille constante

à l'écran. Ainsi, il faut réduire ou amplifier cette fréquence d'un facteur $\alpha = 2^{ld}$. En pratique, il suffit de diviser les coordonnées 3D de la texture volumique par α pour obtenir l'effet désiré.

Pour créer l'illusion de la continuité du zoom, quatre octaves $\Omega_{i=1...4}$, réduites d'un facteur supplémentaire 2^{i-1}, sont mélangées avec leur facteur de pondération respectif $w(s)_{i=1...4}$. Cette pondération, dépendante du paramètre d'interpolation $s = \log 2(distC) - ld$, doit être définie de telle sorte que la transition soit douce lors du décalage des octaves. En pratique, nous avons utilisé les poids suivants :

$$\begin{array}{lcllcl} w(s)_1 & = & 0.6s & w(s)_3 & = & 0.3 - 0.15s \\ w(s)_2 & = & 0.6 - 0.3s & w(s)_4 & = & 0.15 - 0.15s \end{array}$$

Avec ce formalisme, lorsque le facteur de zoom a doublé, $\Omega_{i+1}^{zoom=final} = \Omega_i^{zoom=initial}$: les octaves sont alors automatiquement décalées et s est réinitialisé à 1, débutant ainsi un nouveau cycle.

5.1.2 Résultats et discussion

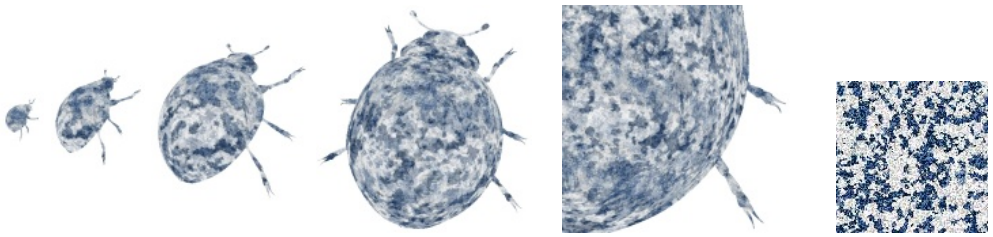


FIG. 5.1: Zoom infini tridimensionnel : texture non-structurée granuleuse (original 2D à droite).

Les Figures 5.1 et 5.2 montrent cinq captures d'écran prises durant le zoom sur deux textures volumiques différentes. La première a été délibérément choisie pour son aspect granuleux non-structuré et la seconde, au contraire, pour son « motif » structuré – avec une seule direction privilégiée – bien distinguable lors du zoom.

FIG. 5.2: Zoom infini tridimensionnel : le « motif » conserve une taille constante à l'écran et reste cohérent au cours du zoom (original 2D à droite).

Il est à noter que l'on retrouve en 3D un mélange de bonne qualité statique et dynamique – au sens des critères que nous avons définis : similarité avec l'original, recouvrement du motif, continuité et illusion de zoom – pour ces deux classes de textures. En effet, dans les deux cas, il est important de remarquer la fréquence globalement constante du mélange quelque soit le niveau de zoom. Par ailleur, sur une vidéo, on se rend pleinement compte de la cohérence temporelle du zoom durant le mouvement. Enfin, si la similarité avec la texture d'origine et la minimisation des recouvrements ne sont pas parfaites, elles sont à un niveau visuellement acceptable pour ces classes de textures.

La figure 5.3 montre une séquence de zoom similaire, pour une texture structurée ne possédant pas de direction privilégiée. Comme dans le cas 2D, force est de constater que le « motif » est brouillé par le mélange des octaves et donc plus difficilement distinguable.

Il est également important de noter que, la texture volumique étant "mappée" en espace objet, elle subit la projection perspective lors du rendu. Ainsi, la déformation perspective du « motif » peut devenir très

FIG. 5.3: Zoom infini tridimensionnel : le « motif » est brouillé par le mélange et de fortes déformations perspectives sont visibles (original 2D à droite).

visible, comme en bordure de la coccinelle sur le gros plan. Cette déformation peut être problématique dans le cas du rendu de certains styles NPR, ceux-ci imitant généralement des techniques traditionnelles travaillant dans un plan 2D. Cette limitation est inhérente à notre solution et constitue le prix à payer pour bénéficier de la cohérence temporelle des textures volumiques. Nous proposerons une étude plus complète de ce problème pour le cas du style aquarelle (cf. section 6.1).

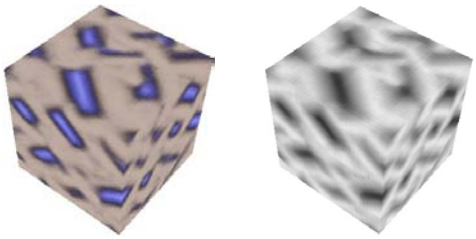
5.2 Pistes pour l'extension de nos autres solutions

Comme en dimension deux, notre méthode par fondu ne s'avère pas très adaptée aux textures structurées à "features" segmentables ou ne possédant pas de direction privilégiée. Nous discuterons donc des extensions, réalisées et envisagables, de notre approche 3D à partir des solutions bidimentionnelles proposées au chapitre 4.

5.2.1 Paramètres de "Dynamic Canvas" et fonction de mélange

La même étude que dans le cas bidimensionnel des paramètres de l'algorithme "Dynamic Canvas" a été réalisée en 3D. Les conclusions auxquelles nous sommes arrivés sont identiques : les deux seuls paramètres modifiables simplement sont le nombre d'octaves – entre trois et quatre pour une continuité temporelle suffisante – et la fonction de mélange – linéaire, gaussienne ou « intelligente ».

Nous avons étendu en 3D notre méthode de mélange « intelligent » utilisant une *feature map*. En effet, à condition de posséder la *feature map* 2D de l'image d'exemple dont on veut synthétiser la texture volumique, nous sommes capables, avec notre implémentation de la méthode de Kopf *et al.*, de générer simultanément la *feature map* volumique associée (fig. 5.4).



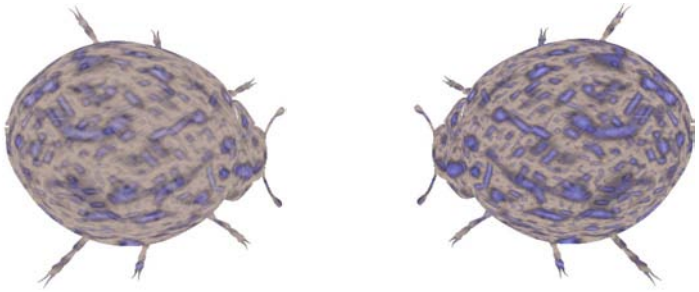
(a) Texture volumique. (b) *Feature map.*

FIG. 5.4: Texture volumique et *feature map* associée synthétisées avec notre implémentation de la méthode de Kopf *et al.*

En utilisant cette information supplémentaire, il nous est possible de rehausser dans le mélange – en modifiant leur pondération en fonction de la *feature map* – le « motif » dans le cas des textures à "features" segmentables sur fond uniforme (fig. 5.5). Cette technique n'empêche cependant pas le recouvrement du « motif » : il serait nécessaire d'étendre la méthode de déformation des octaves au cas tridimensionnel pour pallier ce problème.

5.2.2 Alignement du « motif » par déformation des octaves

Nous n'avons pas étendu au cas tridimensionnel les méthodes d'alignement proposées en dimension deux. Techniquement il n'y a pas d'obstacle majeur à leur passage en 3D : une triangulation volumique de la *feature map* est tout à fait possible comme le calcul de son gradient.



(a) Sans la *feature map*. (b) Avec la *feature map*.

FIG. 5.5: Zoom infini tridimensionnel sur une textures à *features* segmentables sur fond uniforme, en rehaussant (b), ou non (a), le « motif » avec la *feature map*.

Il est cependant beaucoup plus difficile en 3D de définir quel devrait être l'alignement des *features*, car il doit être correct quelle que soit la coupe – pas nécessairement plane – dans la texture volumique. Une étude plus approfondie de cet alignement, en fonction du type de texture, serait, par conséquent, nécessaire pour réaliser cette extension et constituerait une nouvelle piste de recherche future.

5.2.3 Optimisation de la texture originale pour le mélange

Cette dernière approche, qui a montré des résultats prometteurs pour certaines textures 2D, pourrait également définir une autre voie à approfondir. Cependant, plutôt que de considérer cette technique comme un pré-traitement de la texture avant le mélange, il nous semblerait intéressant, dans le cas des textures volumiques, de l'intégrer directement à l'algorithme de synthèse de Kopf *et al.*.

En effet, la synthèse, comme notre méthode, est déjà posée comme un problème d'optimisation. L'objectif serait donc de reformuler ce problème, non plus pour que la texture volumique soit semblable à la texture d'origine, mais pour que le mélange de ses n octaves, à tout instant du zoom infini, soit le plus ressemblant possible à l'image d'exemple 2D.

5.3 Bilan

Comme nous l'avons discuté précédemment, notre solution au problème du zoom infini en 3D n'est pas optimale pour toutes les classes de textures. Certaines limitations actuelles pourraient cependant être levées, en étendant au cas tridimensionnel les techniques proposées en 2D, notamment l'alignement du « motif » par déformation.

Néanmoins, cette solution est généralement visuellement acceptable lorsqu'elle est utilisée comme support d'un style non-photoréaliste, comme nous le montrerons dans le chapitre suivant. En effet, le « motif » de la texture volumique n'est alors pas directement affiché à l'écran, mais entre dans un "pipeline" de rendu plus complexe qui peut grandement modifier sa perception finale.

Chapitre 6

Stylisation et résultats

Ce chapitre a pour but d'illustrer, à travers différent styles non-photoréalistes, la cohérence temporelle apportée par notre solution (se référer aux vidéos disponibles à l'adresse http://artis.imag.fr/Publications/2008/Ben08/ pour des exemples dynamiques). Nous souhaitons non seulement montrer la simplicité avec laquelle différents types de rendus NPR peuvent être obtenus, mais aussi souligner la qualité des visuelle des animations produite par cette approche.

6.1 Aquarelle

Nous nous intéresserons dans un premier temps au rendu aquarelle, en étendant la méthode de Bousseau *et al.* [7] à nos textures volumiques « infiniment zoomables ».

Trois grandes familles d'algorithmes de rendu aquarelle sont distinguables dans la littérature : les modèles physiques [15, 91], les approches par traitements d'image 2D [55, 60] et le plaquage de textures [61].

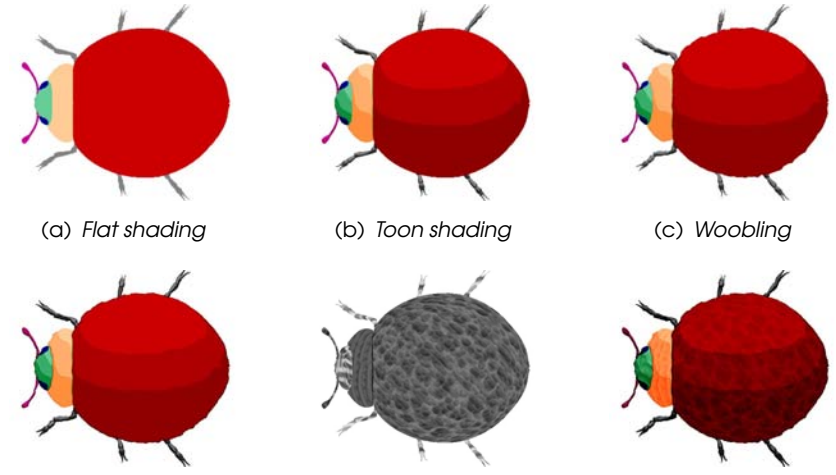


FIG. 6.1: Étapes de notre rendu aquarelle d'après Bousseau *et al.* (7).

La méthode développée par Bousseau *et al.* dans *Interactive watercolor rendering with temporal coherence and abstraction* [7] se classe dans la deuxième catégorie. Elle se base en effet sur une série de traitements d'images 2D pour imiter les effets de l'aquarelle. Nous avons choisi d'étendre cette approche, qui permet simplement d'obtenir une animation aquarelle en temps interactif, à nos textures volumiques. Nous avons pour cela adapté leur "pipeline" de rendu (fig. 6.1).

Le modèle 3D coloré (fig. 6.1(a)) est tout d'abord rendu avec un style *toon* [50] qui produit de larges régions uniformes (fig. 6.1(b)), comparables à celles obtenues par la technique du lavis en aquarelle. Les

oscillations (*woobling*) du bord de ces zones (fig. 6.1(c)) sont ensuite imitées par un décalage des pixels de l'image en fonction du relief de la texture volumique (fig. 6.1(e)). Ces bords sont alors assombris à partir du gradient de l'image (fig. 6.1(d)).

Pour obtenir des couleurs non-homogènes Bousseau *et al.* utilisent deux textures en niveaux de gris : un texture de piments et une texture de turbulences – indiquant la densité des pigments –, ces textures pouvant être générées à partir d'un bruit de Perlin 3D [69]. Ils compositent enfin une texture de papier, comme les deux précédentes, en suivant l'équation empirique :

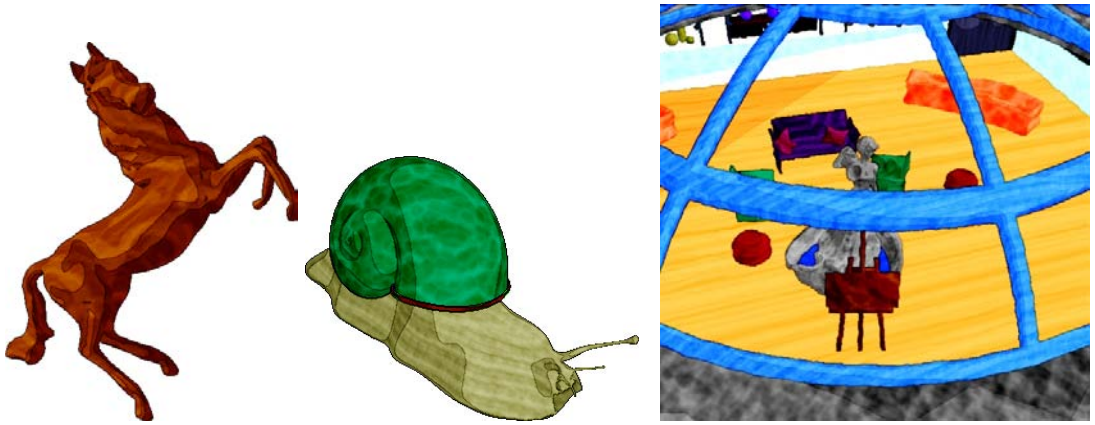
$$C' = C(1-(1-C)(d-1))$$

avec C l'image couleur source (en RGB) et d une densité lue dans la texture de pigments, de turbulences ou de papier.

Nous proposons de remplacer ces trois textures – les deux textures procédurales de pigments et turbulences ainsi que la texture de papier – par une seule texture volumique portant à la fois les informations de pigmentation et de matière (fig. 6.1(e)).

La combinaison de tous ces effets produit le rendu aquarelle final de la figure 6.1(f).

Comme l'illustre la figure 6.2 sur une scène complexe, les textures volumiques offrent une bien plus grande richesse de matériaux et d'effets de pigments que l'approche de Bousseau *et al.*. De plus, ces effets restent totalement cohérents tout au long de l'animation alors que le *woobling* ou la texture de papier ne l'était pas dans leur cas.



(a) Objets simples (b) Scène complexe

FIG. 6.2: Rendu aquarelle avec textures volumiques.

6.2 Collage

Le collage est une technique artistique traditionnelle, accessible dès le plus jeune âge. Pour la pratiquer, il est uniquement nécessaire à l'artiste de disposer de suffisemment de papiers différents pour pouvoir laisser libre cours à sa créativité. Les animations à partir de collages sont cependant assez rares, déplacer les bandes de papier image par image, les redécouper, en ajouter de nouvelles étant des tâches complexes et laborieuses.

Nos textures volumiques semblent bien adaptées pour répondre à ce problème. En effet, nous sommes non seulement capables de synthétiser une riche galerie de textures aux « motifs » variés, mais encore d'assurer la cohérence spatiale et temporelle de ces texture 3D tout au long de l'animation.

Nous n'avons pu trouver dans la littérature que deux publications concernant le rendu « collage » : l'un intégralement en dimension deux [38] et l'autre totalement en 3D [24].

Contrairement à ces deux méthodes, nous proposons une stylisation partant en entrée d'une scène 3D et produisant en sortie une animation dans un style « collage » d'apparence 2D. Notre « collage » trouve son inspiration dans les œuvres des *Collagistes*[1] – technique de création artistique initiée par Georges Braque et Pablo Picasso au XIX^{e} siècle – et les travaux manuels réalisés par découpage/collage de papiers peints en classe de maternelle.

Notre objectif est de laisser une grande liberté à l'artiste quant au choix de l'appariement objet-papier, tout en conservant un mode d'utilisation très simple. Pour illustrer cette idée, nous avons développé un système permettant d'assigner jusqu'à trois textures volumiques par objet – soit trois papiers différents – et de « découper » des bandes dans ces textures en fonction des conditions d'illumination.

Nous avons utilisé un *X-Toon shader* [4] pour permettre une recolorisation cohérente des trois textures. Ce shader prend en entrée une texture 2D (fig. 6.3) : l'axe horizontal correspondant au classique calcul d'éclairage du modèle lambertien – $n.l$ où n est la normale à l'objet et l la direction de la lumière – et l'axe vertical à un certain niveau de détails, dans notre cas, la valeur du niveau de gris de la texture volumique. En peignant simplement les trois bandes de la texture 2D dans un logiciel de dessin (Gimp©, Photoshop©...), l'artiste peut contrôler la couleur qu'il veut donner à son collage pour tout couple éclairage / niveau de gris.

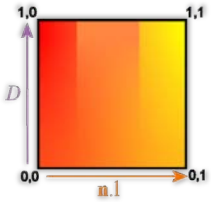


FIG. 6.3: Texture *X-Toon* d'après Barla *et al.* (4).

(a) Objets simples (b) Scène complexe

FIG. 6.4: Rendu collage avec textures volumiques.

Afin de souligner l'aspect papier du collage, nous avons ajouté un traitement 2D des bords des bandes. Cet effet est semblable au *woobling* de l'aquarelle mais avec un liseré blanc supplémentaire pour mettre en avant la découpe voire déchirure du papier.

La figure 6.4 donne un exemple de résultat obtenu avec notre méthode. Il est à noter que la simplification des formes obtenues par le *X-Toon shader*, qui est uniquement basée sur l'éclairage lambertien, n'est pas toujours la mieux adaptée pour mettre en avant la géométrie. D'autres approches, comme celle proposée cette année à NPAR par Vergne *et al.* [94], proposent un stylisation plus respectueuse du relief des objets qui serait intéressante pour notre style « collage ».

[1] http://fr.wikipedia.org/wiki/Collagisme

6.3 Style binaire : stippling, hachures...

De nombreuses techniques ont été proposées pour le rendu par points (*stippling*) [17, 44, 81], de hachures [74] et plus généralement de marques binaires (gravures [67], dessin au crayon ou fusain...).

Une des approches les plus simple pour reproduire ce type d'effet est celle proposée par Durand *et al.* [21]. Elle consiste à obtenir une marque binaire par seuillage d'un champ de hauteur – sous la forme d'une texture en niveaux de gris – en fonction du ton de destination souhaité. (fig. 6.5).

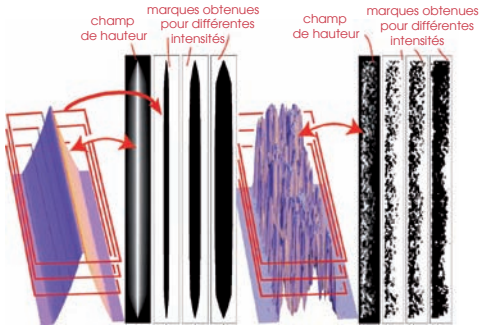


FIG. 6.5: Marques binaires obtenues par seuillage d'un champs de hauteur en fonction du ton de destination, d'après Durand *et al.* (21).

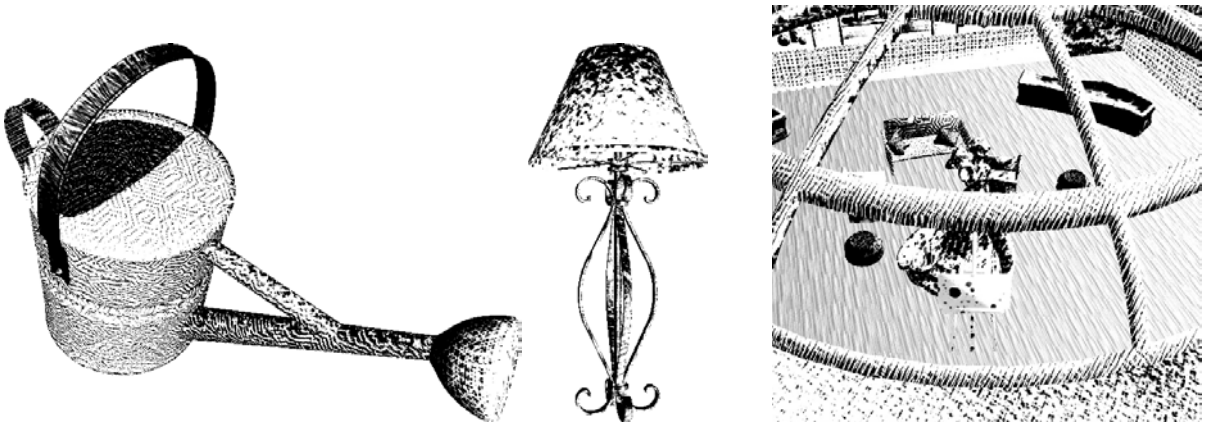
Dans la lignée de cette approche, nous proposons d'utiliser une texture volumique en niveaux de gris comme champ de hauteur. Le seuillage de cette texture est effectué à partir de l'intensité diffuse I de l'équation du modèle lambertien (produit scalaire entre la normale à l'objet et la direction de la lumiere). L'image seuillée peut alors s'exprimer comme : $T_s = S(T - I)$, avec T le champ de hauteur et S la fonction échelon (valant 0 pour $T - I \leq 0$ et 1 pour $T - I > 0$).

Pour pallier au problème de crénelage ("aliasing") et adoucir les transitions entre les marques, nous avons remplacé la fonction échelon par une interpolation linéaire sur un intervalle de taille $2w$:

$$s(t,i,w) = \begin{cases} 1 & t-i < -w \\ 1/2 - (t-i)/2w, & -w \leq t-i \leq w \\ 0 & t-i > w \end{cases}$$

Par défaut nous avons fixé $w = 0.1$ dans nos exemples.

La figure 6.6 illustre la diversité des styles binaires combinables dans une seule image, en attribuant simplement des textures différentes à chaque objet. Il est également à noter que la méthode de zoom infini s'adapte parfaitement à ce type de rendu, les nouvelles marques binaires apparaissant progressivement pour maintenir le ton défini par l'éclairage.

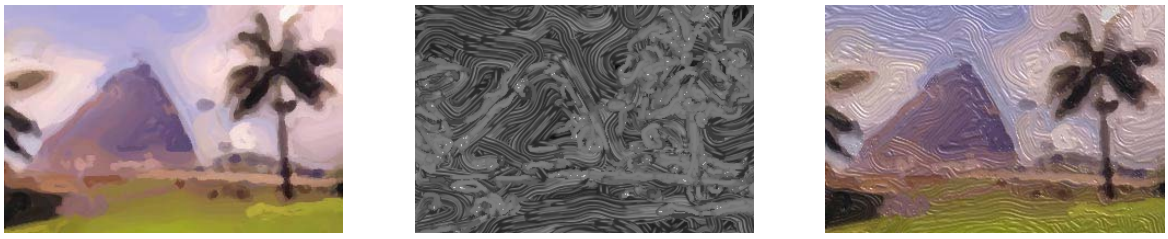


(a) Objets simples (b) Scène complexe

FIG. 6.6: Rendu binaire avec textures volumiques.

6.4 "Fast Paint Texture 3D"

Nous proposons enfin un style « peinture » directement inspiré de l'approche d'Hertzmann dans *Fast Paint Texture* [34], plusieurs fois réutilisée, par exemple par Tateosian *et al.* [87].



(a) Peinture originale (b) Champ de hauteur (c) Peinture ré-illuminée

FIG. 6.7: *Fast Paint Texture* d'Hertzmann (34).

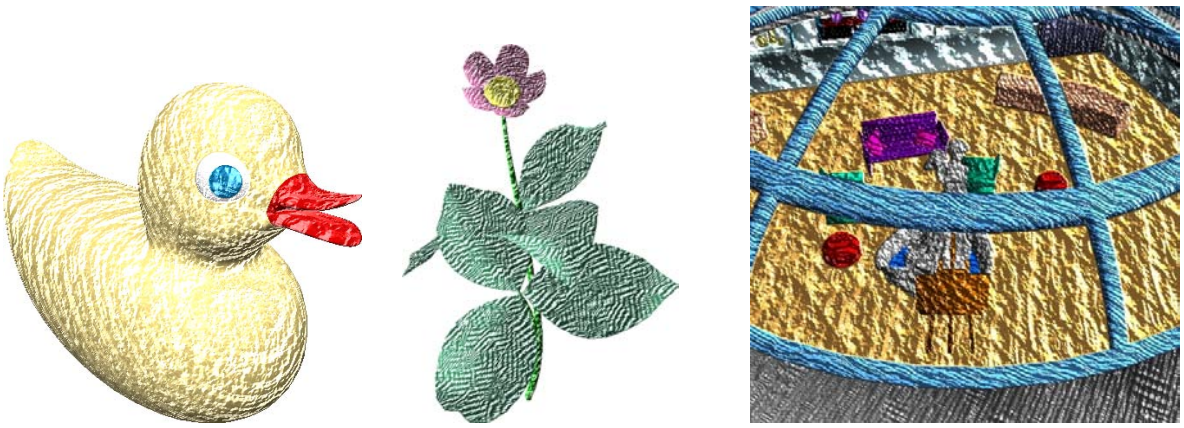
Son idée est de simuler visuellement l'apparence des coups de pinceaux d'une peinture sous une certain illumination. Son algorithme prend en entrée une liste ordonnée de coups de pinceaux, un modèle d'illumination et une carte de hauteur pour chaque type de coups de pinceau. Une « peinture » est produite en sortie, après trois étapes de traitement successives :

- une image couleur est obtenue en superposant, dans l'ordre, les coups de pinceau (fig. 6.7(a)),
- un champ de hauteur est calculé en superposant, cette fois, la carte de hauteur de chaque marque (fig. 6.7(b)),
- la peinture finale est obtenue par placage de relief (*bump-mapping*)[2] sous le modèle d'illumination de Phong (fig. 6.7(c)).

Nous proposons d'étendre ce principe à nos textures volumiques en considérant leur projection comme une carte de hauteur. Chaque objet de la scène se voit assigner une couleur et une texture 3D. Puis, en suivant les trois étapes de l'algorithme d'Hertzmann :

- une image couleur est générée par rendu Gouraud de la scène, sans tenir compte des textures,
- un champ de hauteur est calculé par rendu de la scène sans éclairage en ne tenant compte que des textures,
- la « peinture » finale est obtenue par placage de relief en utilisant les deux images précédentes.

La figure 6.8 propose un exemple obtenu avec cette approche.



(a) Objets simples (b) Scène complexe

FIG. 6.8: "Fast Paint Texture 3D".

[2] Le *bump-mapping* consiste à perturber les normales de l'objet en fonction des dérivés directionnelles du champ de hauteur.

Chapitre 7

Conclusion et travaux futurs

La principale contribution de ce travail est de proposer une nouvelle solution au problème récurrent en rendu non-photoréaliste de la cohérence temporelle au cours de l'animation d'une scène 3D. Cette solution se base sur l'utilisation de textures volumiques, traduisant le médium utilisé pour styliser le remplissage des objets 3D, auxquelles nous avons données la propriété d'être « infiniment zoomable ».

Ces textures volumiques – aujourd'hui facilement synthétisables depuis une image d'exemple 2D – garantissent la cohérence spatiale et temporelle quels que soient les mouvements de la caméra ou des objets de la scène, tout en autorisant une vaste palette d'effets et de « motifs » jusqu'alors difficile à obtenir. En outre, elles sont d'une utilisation aisée, permettent de créer des animations interactives et pourraient donc potentiellement aussi bien être intégrées dans un logiciel de rendu hors-ligne (Mental Ray©, RenderMan©...) que dans un moteur de jeu vidéo.

Plusieurs améliorations peuvent néanmoins être apportées aux méthodes proposées. Le zoom infini par mélange d'octaves autosimilaires reste perfectible pour certaines classes de textures, en particulier les plus structurées. Les algorithmes de préservation du contraste, d'optimisation et de déformation des octaves que nous avons développés nécessiteraient de plus amples recherches pour en améliorer les résultats et proposer leur extension au cas tridimensionnel. En particulier, une meilleure méthode d'optimisation visant à véritablement extraire l'autosimilarité du « motif » de la texture – à l'image des méthodes de compression fractale, par exemple – pourrait être une voie pertinente à suivre.

D'autres approches semblent également intéressantes à explorer. Han et *al.* dans *Multiscale Texture Synthesis* [30] (accepté à SIGGRAPH cette année) proposent notamment une méthode de synthèse dynamique multi-échelle de textures 2D – étendant les travaux de Lefebvre et Hoppe [54] – qui permet de créer un effet de zoom infini non cyclique (fig. 7.1). Le problème est cependant d'étendre cette approche au cas tridimensionnel. Une possibilité envisageable est d'adapter la nouvelle méthode de synthèse volumique à la surface des objets de Dong et *al.* [19] (présentée prochainement à EGSR).

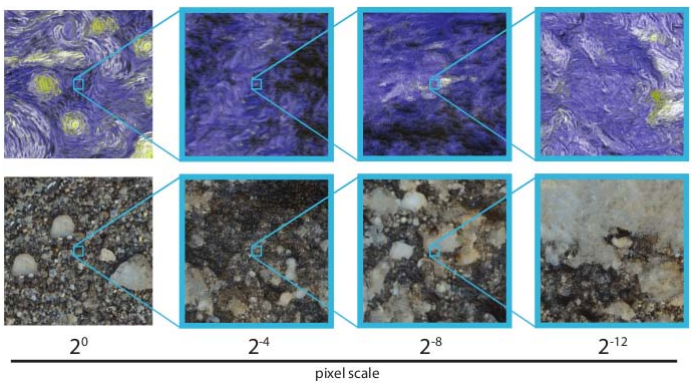


FIG. 7.1: Zoom infini bidimentionnel dynamique par la méthode de Han et *al.* (30)

De plus, dans le cas des textures à “features” segmentables, des approches à bases de distributions de points « infiniment zoomables » – s’inspirant des travaux de Kopf et *al.* [44] et de Lagae et Dutré [49] – pourraient également constituer des pistes de recherche intéressantes. La difficulté est cependant de segmenter puis définir automatiquement la géométrie 3D des “features” d’une texture 2D, ainsi que de déterminer la distribution de ces “features”.

Enfin, comme souligné lors de l’état de l’art sur la cohérence temporelle, l’absence d’étude formalisant et quantifiant perceptuellement ce problème majeur du NPR est une importante lacune. Qu’il s’agisse d’une texture ou d’une distribution de points, il semble nécessaire d’apporter de nouveaux outils d’analyse de ce phénomène inhérent à toute animation stylisée, puis de traitement de ses artefacts.

Annexe A

Synthèse de textures volumiques

Si l'utilisation de textures afin d'ajouter des détails, trop complexes pour être définis géométriquement, est une pratique courante en rendu, elle fait essentiellement appel aux textures 2D. Par conséquent, la majorité des algorithmes de synthèse de textures a été développée pour le cas bidimensionnel.

Cependant, avec la croissance de la mémoire disponible sur les cartes graphiques, les textures volumiques tendent à se généraliser. La difficulté majeure posée par les textures volumiques reste leur acquisition, dans la mesure où digitaliser un matériau en trois dimensions est très difficile, voire généralement impossible.

Des solutions, procédurales dans un premier temps et à partir d'exemples dans un second, ont néanmoins été proposées pour résoudre ce problème. Dans cette annexe, nous les présenterons et discuterons de leurs avantages et défauts respectifs (cf. section A.1), avant de choisir l'une d'entre-elles et d'en détailler l'implémentation que nous en avons faite (cf. section A.2).

A.1 État de l'art

Nous n'effectuerons pas dans cette section un état de l'art complet des techniques de synthèse de texture, mais un rapide panorama des grandes classes de méthodes ayant souvent été proposées en 2D et étendues en 3D, en mettant en avant leurs qualités et défauts respectifs afin de justifier notre choix pour l'une d'entre-elles.

A.1.1 Textures procédurales

Dès 1984, Gardner propose d'utiliser des textures 3D procédurales – sous forme de sommes et produits de fonctions sinusoïdales 2D, modulées en phase et en amplitude selon l'axe z – pour synthétiser des paysages et des nuages [25].

Mais c'est seulement l'année suivante que les textures volumiques apparaissent véritablement sous cette dénomination, dans les travaux de Peachey [69] et Perlin [71]. Peachey propose une extension de certaines méthodes procédurales 2D au cas 3D tandis que Perlin introduit un bruit blanc 3D, modulé par une fonction de perturbation. Avec ces deux méthodes, des matériaux tels que le granite ou le bois ont pu être générés (fig. A.1).

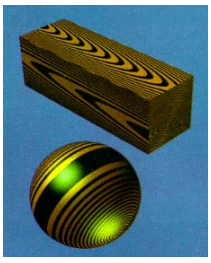


FIG. A.1: Texture volumique procédurale de bois par Peachey (69).

Les avantages de ces approches procédurales sont la compacité de leur représentation – « quelques » lignes de code –, leur indépendance vis à vis de la résolution de l'image en sortie et leur modularité. Leur inconvénient majeur est la nécessaire expertise de l'utilisateur devant définir, à l'aveugle, une combinaison de paramètres peu intuitifs susceptibles de produire le résultat escompté.

Des produits commerciaux (*Allegorithmic*©, *Genetica*©...) proposent des solutions pour simplifier la tâche de l'utilisateur, mais à l'heure actuelle uniquement dans le cas bidimensionnel. Des travaux proposent également d'inférer les paramètres depuis une image d'exemple : dans le cas de textures 2D présentant un motif régulier [52] ou, en 2D et 3D, en utilisant des algorithmes génétiques [37, 76]. Dans

tous les cas, le code de la texture procédurale doit préalablement avoir été défini, seuls les paramètres restant à déterminer.

Par conséquent, les approches procédurales 3D sont encore d'un usage assez restreint, les méthodes de synthèse à partir d'exemples étant privilégiées pour leur simplicité d'utilisation.

A.1.2 Synthèse à partir d'exemple(s)

Cette classe de méthodes utilise un exemple de la texture à synthétiser – généralement une ou plusieurs coupes 2D pouvant être scannées ou facilement « peintes » par l'utilisateur – et génère automatiquement une texture volumique similaire à ce modèle. Ce problème est particulièrement difficile dans la mesure où l'exemple n'apporte qu'une information partielle sur la texture volumique, cette information devant donc être, d'une façon ou d'une autre, complétée.

Deux classes de méthodes ont été proposées ces vingt dernières années pour répondre à ce problème : les approches paramétriques et celle non-paramétriques.

Approches paramétriques

Cette première classe d'approche consiste à construire un modèle paramétrique global de la texture à partir d'une image d'exemple.

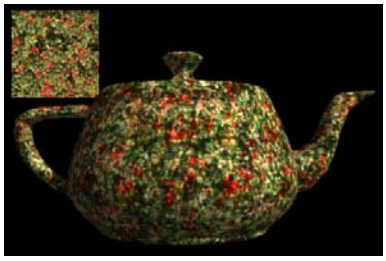


FIG. A.2: Texture volumique synthétisée par la méthode de Heeger et Berger (32).

Ghazanfarpour et Dischler [26] furent les premiers à proposer une telle approche. Ils analysent spectralement par FFT – transformée de Fourier rapide – l'image exemple pour en extraire une fonction de base et une fonction de perturbation, semblables à celles de Perlin. A partir de ces fonctions, la synthèse de textures volumiques est possible, même si le contrôle selon la troisième dimension est difficile et le temps de calcul prohibitif.

Heeger et Berger [33] proposent une méthode de décomposition pyramidale (laplacienne ou *steerable*) pour les textures isotropes (sans direction privilégiée). Partant d'un cube de bruit blanc, ils le font converger itérativement à chaque niveau de la pyramide vers l'image exemple par mise en correspondance de leur histogramme. Les résultats obtenus (fig. A.2) sont assez convaincants mais restreints à une petite classe de textures (isotropes et peu structurées).

Dishler *et al.* [18] combinent ces deux méthodes pour traiter le cas des textures anisotropes, en utilisant deux images d'exemple. Cette approche reste toutefois inadaptée pour synthétiser des textures présentant de petites structures : la paramétrisation globale ne parvenant pas à intégrer ces variations fines.

Approches non-paramétriques

Les approches non-paramétriques ont largement été développées en 2D avant d'être étendues au cas tridimensionnel. Elles ont en commun l'idée de faire grandir la texture synthétisée pixel (ou groupe de pixels) par pixel à partir de l'image d'exemple. Quatre classes de méthodes se dégagent de la littérature : les méthodes probabilistes, pixelliques, par régions et par optimisation.

- Méthodes probabilistes

FIG. A.3: Hypothèse de De Bonet (6) : en dessous d'une certaine résolution, la permutation de certaines régions est indiscernable (régions encadrées).

L'hypothèse de De Bonnet [6] est, qu'en-dessous d'une certaine résolution, certaines régions sont perceptuellement indiscernables les unes des autres et, de part le caractère aléatoire de la texture, sont

interchangeables (fig. A.3). Il propose donc de décomposer l'image d'exemple et la texture 2D à synthétiser sous forme d'une pyramide laplacienne. La pyramide de la texture est alors remplie à chaque niveau par échantillonnage de la pyramide exemple, sous contraintes locales de similarité. Cette méthode ne parvient cependant pas à synthétiser les textures très structurées et n'a pas été directement étendue à la dimension supérieure.

La méthode proposée en 2007 par Qin et Yang [77] se rapproche cependant de cette approche probabiliste. Elle propose en effet d'utiliser trois images d'exemple, une pour chaque coupe axiale du cube, et de les décomposer en *Basic Gray Level Aura Matrices* (BGLAMs).La BGLAM traduit la distribution de la probabilité de cooccurrence des niveaux de gris d'une image pour tout déplacement. Partant d'un cube de bruit blanc et de ces matrices – une pour chaque canal de couleur et pour chaque coupe –, un échantillonnage aléatoire est itérativement effectué tant que la distance entre le cube synthétisé et les BGLAMs diminue (fig. A.4).

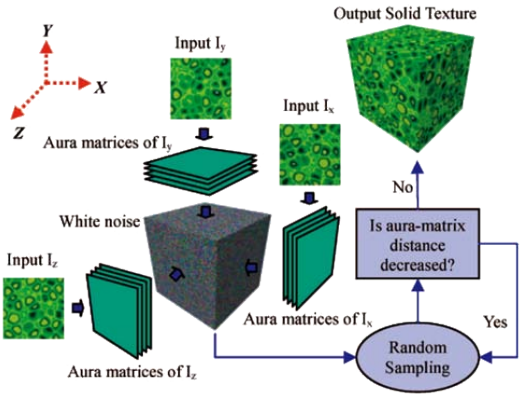


FIG. A.4: Principe schématisé de l'algorithme de synthèse de Qin et Yang (77).

Bien que les résultats obtenus par cette approches soient convainquant (fig. A.9(b)) deux problèmes se posent. Pour traiter des textures en couleur, il est nécessaire de décorréler les trois canaux et de les synthétiser séparément, ce qui peut induire des artefacts. La construction des BGLAMs est très coûteuse – croissance quadratique de cette structure – et ne garantie pas nécessairement une information statistique suffisante.

Un dernière approche statistique a été proposée par Jagnow *et al.* [39] pour la synthèse de textures volumiques de matériaux à base de particules emprisonnées dans un liant. Elle se fonde sur des techniques stéréologiques (analyse morphométrique des structures spatiales basée sur des échantillonnages par coupes) pour estimer la distribution des particules. Les résultats obtenus avec cette méthode sont très fidèles à l'image d'exemple (fig. A.5). Ils requièrent néanmoins la modélisation 3D manuelle des différents types de particules par un utilisateur : tâche longue et contraignante.

FIG. A.5: Texture volumique synthétisée par la méthode de Jagnow *et al.* (39).

- Méthodes pixelliques

Les méthodes pixelliques, introduites par Efros et Leung [23] en 1999, font converger un bruit blanc vers une texture ressemblant à l'image exemple en le modifiant pixel après pixel. La figure A.6 illustre ce processus dans le cas de l'algorithme de Wei [96] qui fixe un parcours dans l'ordre de rasterisation (de haut en bas et de gauche à droite).

FIG. A.6: Algorithme de synthèse de Wei (96).

Pour déterminer la valeur du pixel p à synthétiser, le voisinage spatial $N(p)$ en forme de L est comparé à tous les voisinages $N(p_i)$ de l'image exemple. Le pixel d'entrée p_i avec le voisinage $N(p_i)$ le plus similaire (en norme L_2) à $N(p)$ est assigné à p. Cette approche vise ainsi à maintenir localement, autant que possible, la similarité entre l'exemple et la texture synthétisée.

De nombreuses extensions ont été apportées à cet algorithme de base : schéma multi-résolution, structures accélératrices de recherche du meilleur voisinage [97], contraintes additionnelles sur la synthèse [3, 53], changement de l'espace de recherche [54], synthèse à la surface des objets [89] et extension à la 3D (jusqu'à trois images d'exemples).

Les résultats obtenus en 3D avec ce type d'approches présentent cependant de nombreux artefacts : en particulier, une tendance au flou importante et l'incapacité de préserver les petites structures (fig. A.9(c)).

- Méthodes par régions

FIG. A.7: *Lapped Textures* de Praun *et al.* (73).

L'extension logique des méthodes pixelliques est l'utilisation d'un ensemble de pixels – un "patch" de l'image d'exemple – pour synthétiser une nouvelle texture, ressemblant toujours plus à l'original. Plusieurs approches [22, 48, 56] ont été développées dans le cas bidimensionnel pour déterminer l'arrangement de ces régions, leur découpe dans le modèle et leur recouvrement optimal.

Une méthode de synthèse à la surface des objets, les *Lapped Textures*, a été proposée par Praun *et al.* [73]. Elle propose de copier les "patchs" de texture directement à la surface de l'objet en suivant un champ de vecteurs tangentiels (fig. A.7). Ce champ de vecteurs, spécifié par l'utilisateur, permet le contrôle local de la taille et l'orientation de la texture.

Cette méthode admet cependant plusieurs limitations. Le bord des "patchs" devient visible pour les textures ayant une composante basse fréquence importante. Des artefacts peuvent apparaître autour des points de singularité du champ de vecteur. Pour les textures très structurées, un effet de flou ou de mauvais alignement des structures peut être visible. Enfin, l'implémentation et l'utilisation de cette technique n'est pas évidente.

FIG. A.8: Texture volumique synthétisée par la méthode de Takayama *et al.* (85).

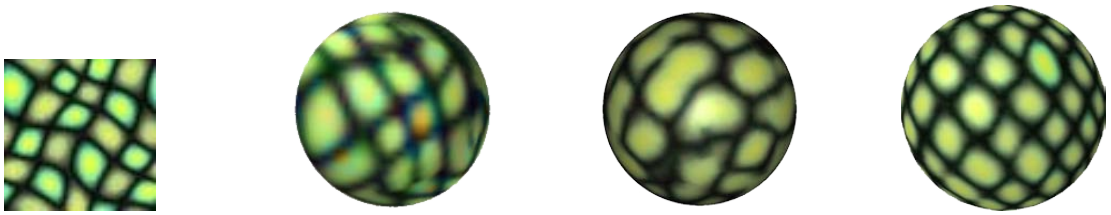
Une extension de cette approche aux textures volumiques, par Takayama *et al.* [85], est à paraître cette année à SIGGRAPH. Elle est capable de synthétiser des textures anisotropes – ayant jusqu'à deux directions privilégiées – en suivant un champ de vecteur volumique défini par l'utilisateur (fig. A.8). Cette technique nécessite cependant en entrée, non pas une image 2D, mais un "patch 3D". L'acquisition – manuelle ou par digitalisation – de ce volume initial est une tâche complexe, qui limite l'utilisabilité de cette méthode pour un grand nombre de textures. En outre, elle hérite des limitations des *Lapped Textures*.

- Méthodes par optimisation

Cette dernière classe de méthodes non-paramétriques considère que la texture à synthétiser doit évoluer comme un tout. L'objectif du processus d'optimisation est de minimiser une fonction d'énergie globale mesurant la proximité entre la texture synthétisée et l'image d'exemple.

Kwarta *et al.* [46] proposent de définir cette similarité globale comme la combinaison des similarités locales des voisinages de tout pixel de la texture. Ils font ainsi l'hypothèse que maximiser la similarité locale est une condition suffisante pour garantir la similarité globale entre la texture 2D et l'image d'exemple. L'énergie ainsi définie est alors optimisée par un algorithme de type Espérence-Maximisation (EM).

Cette approche a été étendue au cas tridimensionnel "2D + temps" par Wexler *et al.* pour supprimer des éléments dans une vidéo [98] puis aux textures volumiques par Kopf *et. al* [45]. Nous avons choisi de ré-implémenter cette dernière méthode aux regards de la qualité et de la diversité des résultats qu'elle permet de produire (fig. A.9(d)) et de sa simplicité d'utilisation (une seule image d'exemple pouvant être suffisante).



(a) image d'exemple. (b) méthode statistique de Qin et Yang (77). (c) méthode pixellique de Wei (96). (d) méthode d'optimisation de Kopf *et al.* (45).

FIG. A.9: Textures volumiques, synthétisées depuis le même exemple (a), par différentes méthodes.

A.2 Implémentation de *Solid Texture Synthesis from 2D Exemplars*

L'objectif poursuivi par Kopf *et al.* dans *Solid Texture Synthesis from 2D Exemplars* [45] est, connaissant en entrée une ou plusieurs coupes axiales d'un volume de texture, de produire en sortie un cube de texture volumique similaire à ces images.

La méthode qu'ils proposent combine des idées et techniques utilisées en synthèse de textures paramétriques et non-paramétriques. Elle est basée sur une optimisation locale, selon les trois axes du cube, guidée par des informations statistiques tirées des images d'exemple.

A.2.1 Détails

La méthode de Kopf *et al.* étend l'optimisation globale de Kwarta *et al.* [47] à la 3D en lui adjoignant une étape de mise en correspondance d'histogrammes, inspirée des travaux d'Heeger et Berger [33]. L'ajout de cette information statistique globale permet une convergence plus rapide de l'optimisation – en autorisant l'utilisation d'un voisinage restreint (8×8) – et lui évite de tomber dans des minimums locaux.

- Énergie à minimiser

En tant que problème d'optimisation, il est nécessaire de définir l'énergie globale à minimiser. Celle-ci correspond à la somme des distances L_2 entre chaque voisinage du cube synthétisé et l'image exemple. Plus formellement, e dénotant l'ensemble des images d'exemple (pouvant être restreint à une seule image) et s la texture volumique à synthétiser, elle s'exprime par la formule :

$$E(s,\{e\}) = \sum_{v} \sum_{i \in \{x,y,z\}} ||s_{v,i} - e_{v,i}||^r$$

où (fig. A.10) :

- s_v désigne le v^e voxel du cube en cours de synthèse,
- $s_{v,x}$, $s_{v,y}$ et $s_{v,z}$ sont les voisinages (sous forme de vecteurs) de s_v dans les coupes orthogonales aux axes x, y et z,
- $e_{v,i}$ est le plus proche voisinage dans l'image exemple en norme L_2 de $s_{v,i}$,
- l'exposant $r = 0.8$ rend l'optimisation plus robuste.

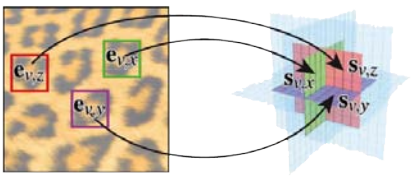


FIG. A.10: Voisinages utilisées.

Cette énergie est alors minimisée itérativement, de façon multi-échelle, en alternant deux étapes : une phase d'optimisation et une phase de recherche du plus proche voisin.

- Optimisation

La phase d'optimisation consiste à mettre à jour, un à un, les voxels s_v en fonction de leurs voisins dans l'image d'exemple (selon les trois axes x, y et z) et d'un schéma de pondération. La valeur optimale

de ces voxels est en définitive une simple moyenne pondérée de trois voisinages différents pris dans l'image d'exemple.

Pour intégrer la mise en correspondance des histogrammes, le schéma de pondération est modifié de telle sorte que les voisins éloignant l'histogramme de la texture volumique de celui de l'image d'exemple soient pénalisés.

Ainsi, un compromis est trouvé entre la similarité locale apportée par les voisinages et la similarité globale induite par l'information statistique des histogrammes.

- Recherche du plus proche voisin

L'objectif de cette deuxième phase est de trouver pour chaque $s_{v,i}$ le texel dans l'image d'exemple $e_{v,i}$ le plus proche. Il s'agit d'un problème classique de recherche du plus proche voisin dans un espace de grand dimension (192 pour un voisinage $8 \times 8 \times 8$).

Suivant les conseils Kopf *et al.*, nous avons utilisé la librairie ANN – *Approximate Nearest Neigbor library* [64] – pour effectuer une recherche approximative, plus efficace à une telle dimension.

Afin d'améliorer la recherche approximative en terme de qualité et de rapidité, nous avons également mise en place une réduction de la dimension par PCA – *Principal Components Analysis* [35, 54, 56] – passant ainsi de la dimension 192 à environ 50, en fonction du type de texture.

Pour accélérer encore cette étape, Kopf *et al.* ont remarqué qu'il n'est pas nécessaire d'effectuer la recherche en tout voxel : n'en considérer qu'un sur deux selon chaque axe est en fait suffisant.

(a) Sans *feature map*. (b) Avec *feature map*.

FIG. A.11: Textures volumiques sythétisées par Kopf *et al.* avec et sans *feature map*.

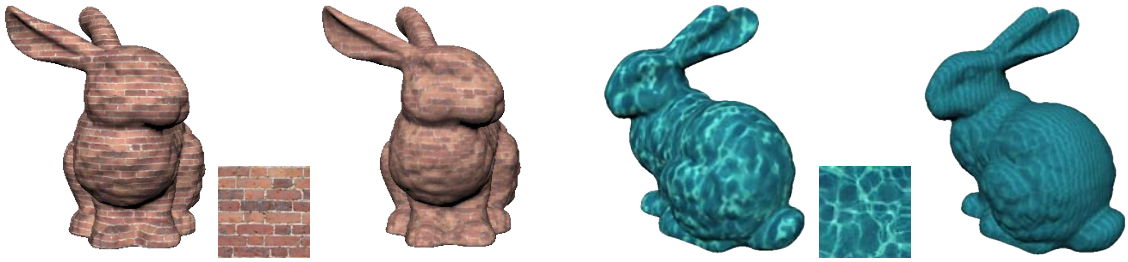
Enfin, pour les textures très structurées, il est possible d'ajouter un canal supplémentaire sous la forme d'une *features map* [54, 99] encodant la distance entre les centres des éléments significatifs. La *features map* guide la recherche du plus proche voisin en proposant une segmentation douce de l'image d'exemple. Cette information supplémentaire améliore très nettement la qualité de ce type de textures, comme l'illustre la figure A.11.

A.2.2 Résultats et limitations

La figure A.12 montre deux exemples de textures synthétisées par Kopf *et al.* et par notre implémentation de leur méthode. Si nos résultats ne sont pas strictement identiques – car nous ne connaissons pas exactement l'ensemble de leurs paramètres et n'avons pas implémenté toutes leurs extensions (voir ci-après) –, ils sont visuellement satisfaisants.

Nous pouvons cependant constater sur la figure A.12(a) que notre texture est moins contrastée, plus floue que celle de Kopf *et al.*. Ils ont en effet résolu ce problème en suivant une technique utilisée par Wexler *et al.* [98].

Durant la phase d'optimisation, ils déterminent par "clustering" (algorithme du *Mean Shift* [12]), pour chaque voxel s_v, les pixels dominants dans les voisinages $e_{v,i}$ et ne calculent la moyenne que sur ces voisins. En diminuant le nombre de voisins retenus au cours des itérations, le moyennage des couleurs est ainsi de plus en plus limité. Faute de temps et étant donné la complexité de cet algorithme supplémentaire, nous n'avons pas implémenté cette amélioration.



(a) Texture de briques. (b) Texture de caustiques.

FIG. A.12: Textures volumiques sythétisées par Kopf *et al.* (gauche) et par notre implémentation (droite).

Il nous faut également reconnaître, au regard de la figure A.12(b), que nos textures ont parfois tendances à suivre une direction privilégiée, alors que la méthode devrait être totalement isotrope (avec une seule image exemple). Nous n'avons cependant pas réussi à déterminer la cause de ce problème dans notre implémentation, l'ordre de parcours des voxels étant réalisé de façon aléatoire.

La figure A.13 illustre un dernier problème de notre implémentation. Il arrive parfois que l'une des couleurs, présente dans l'image d'exemple, disparaisse lors de la synthèse. Nous ne sommes pas parvenus à trouver l'origine de ce phénomène mais deux pistes seraient à explorer : soit une erreur subsiste dans notre schéma de pondération durant la phase d'optimisation (notre histogramme divergeant clairement de celui du modèle), soit la recherche du plus proche voisin n'est pas totalement correcte.

FIG. A.13: Texture volumique synthétisée avec notre implémentation : disparition d'une couleur de l'exemple.

En dépit des deux problèmes mentionnés précédemment, notre implémentation nous a permis de synthétiser avec une bonne précision une grande diversité de textures volumiques, dans les limites de cet algorithme : *i.e.,* en ayant au mieux un contrôle sur une direction privilégiée (en utilisant deux images d'exemple).

Bibliographie

[1] «Vistex database», URL http://vismod.media.mit.edu/vismod/imagery/VisionTexture/vistex.html. 17

[2] Agarwala, A., M. Dontcheva, M. Agrawala, S. Drucker, A. Colburn, B. Curless, D. Salesin et M. Cohen. 2004, «Interactive digital photomontage», *ACM Trans. Graph.*, vol. 23, no. 3, doi :http://doi.acm.org/10.1145/1015706.1015718, pp. 294–302, ISSN 0730-0301. URL http://grail.cs.washington.edu/projects/photomontage/. 29

[3] Ashikhmin, M. 2001, «Synthesizing natural textures», dans *I3D '01 : Proceedings of the 2001 symposium on Interactive 3D graphics*, ACM, New York, NY, USA, ISBN 1-58113-292-1, pp. 217–226, doi :http://doi.acm.org/10.1145/364338.364405. URL http://www.cs.utah.edu/~michael/ts/. 45

[4] Barla, P., J. Thollot et L. Markosian. 2006, «X-toon : An extended toon shader», dans *International Symposium on Non-Photorealistic Animation and Rendering (NPAR)*, ACM. URL http://artis.inrialpes.fr/Publications/2006/BTM06a. 37

[5] Battiato, S., G. Gallo et S. Nicotra. 2003, «Perceptive visual texture classification and retrieval», *Image Analysis and Processing, 2003.Proceedings. 12th International Conference on*, doi :10.1109/ICIAP.2003.1234103, pp. 524–529. 18

[6] Bonet, J. S. D. 1997, «Multiresolution sampling procedure for analysis and synthesis of texture images», dans *SIGGRAPH '97 : Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, ISBN 0-89791-896-7, pp. 361–368, doi :http://doi.acm.org/10.1145/258734.258882. 43

[7] Bousseau, A., M. Kaplan, J. Thollot et F. Sillion. 2006, «Interactive watercolor rendering with temporal coherence and abstraction», dans *International Symposium on Non-Photorealistic Animation and Rendering (NPAR)*, ACM, doi :http://doi.acm.org/10.1145/1124728.1124751. URL http://artis.imag.fr/Publications/2006/BKTS06. 8, 35

[8] Bousseau, A., F. Neyret, J. Thollot et D. Salesin. 2007, «Video watercolorization using bidirectional texture advection», dans *SIGGRAPH '07 : ACM SIGGRAPH 2007 papers*, ACM, New York, NY, USA, p. 104, doi :http://doi.acm.org/10.1145/1275808.1276507. URL http://artis.imag.fr/Publications/2007/BNTS07/. 8

[9] Breslav, S., K. Szerszen, L. Markosian, P. Barla et J. Thollot. 2007, «Dynamic 2d patterns for shading 3d scenes», *ACM Transaction on Graphics (Proceedings of SIGGRAPH 2007)*, vol. 26, no. 3, doi :http://doi.acm.org/10.1145/1276377.1276402, p. 20. URL http://artis.imag.fr/Publications/2007/BSMBT07. 7

[10] Brinkmann, R. 1999, *The art and science of digital compositing*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, ISBN 0-12-133960-2. 29

[11] Coconu, L., O. Deussen et H.-C. Hege. 2006, «Real-time pen-and-ink illustration of landscapes», dans *NPAR '06 : Proceedings of the 4th international symposium on Non-photorealistic animation and rendering*, ACM, New York, NY, USA, ISBN 1-59593-357-3, pp. 27–35, doi :http://doi.acm.org/10.1145/1124728.1124734. 7

[12] Comaniciu, D. et P. Meer. 2002, «Mean shift : A robust approach toward feature space analysis», *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, doi :http://doi.ieeecomputersociety.org/10.1109/34.1000236, pp. 603–619, ISSN 0162-8828. 8, 47

[13] Cornish, D., A. Rowan et D. Luebke. 2001, «View-dependent particles for interactive non-photorealistic rendering», dans *GRIN'01 : No description on Graphics interface 2001*, Canadian Information Processing Society, Toronto, Ont., Canada, Canada, ISBN 0-9688808-0-0, pp. 151–158. 4

[14] Cunzi, M., J. Thollot, S. Paris, G. Debunne, J.-D. Gascuel et F. Durand. 2003, «Dynamic canvas for immersive non-photorealistic walkthroughs», dans *Proc. Graphics Interface*, A K Peters, LTD. URL http://artis.imag.fr/Membres/Joelle.Thollot/dynamic_canvas/. 2, 7, 12, 31

[15] Curtis, C. J., S. E. Anderson, J. E. Seims, K. W. Fleischer et D. H. Salesin. 1997, «Computer-generated watercolor», dans *SIGGRAPH '97 : Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, ISBN 0-89791-896-7, pp. 421–430, doi :http://doi.acm.org/10.1145/258734.258896. 35

[16] Dencker, R. «Image processing library 98», URL http://www.mip.sdu.dk/ipl98/. 24

[17] Deussen, O., S. Hiller, C. van Overveld et T. Strothotte. 2000, «Floating points : A method for computing stipple drawings», dans *Computer Graphics Forum*, vol. 19, pp. 40–51. URL citeseer.ist.psu.edu/deussen00floating.html. 4, 38

[18] Dischler, J. M., D. Ghazanfarpour et R. Freydier. 1998, «Anisotropic solid texture synthesis using orthogonal 2d views», *Computer Graphics Forum*, vol. 17, no. 3, doi :10.1111/1467-8659.00256, pp. 87–95. URL http://www.blackwell-synergy.com/doi/abs/10.1111/1467-8659.00256. 43

[19] Dong, Y., S. Lefebvre, X. Tong et G. Drettakis. 2008, «Lazy solid texture synthesis», dans *Eurographics Symposium on Rendering*. URL http://www.dongallen.com/. 40

[20] Drobisch, M. 1855, «Über musikalische tonbestimmung und temperatur..», *Abhandlungen der Mathematisch-Physischen Klasse der Sächsischen Akademie der Wissenschaften*, p. 2. 10

[21] Durand, F., V. Ostromoukhov, M. Miller, F. Duranleau et J. Dorsey. 2001, «Decoupling strokes and high-level attributes for interactive traditional drawing», dans *Proceedings of the 12th Eurographics Workshop on Rendering Techniques*, Springer-Verlag, London, UK, ISBN 3-211-83709-4, pp. 71–82. URL http://people.csail.mit.edu/fredo/PUBLI/Drawing/. 38

[22] Efros, A. A. et W. T. Freeman. 2001, «Image quilting for texture synthesis and transfer», dans *SIGGRAPH '01 : Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, ISBN 1-58113-374-X, pp. 341–346, doi :http://doi.acm.org/10.1145/383259.383296. 45

[23] Efros, A. A. et T. K. Leung. 1999, «Texture synthesis by non-parametric sampling», dans *IEEE International Conference on Computer Vision*, Corfu, Greece, pp. 1033–1038, doi :10.1109/ICCV.1999.790383. URL http://graphics.cs.cmu.edu/people/efros/research/EfrosLeung.html. 44

[24] Gal, R., O. Sorkine, T. Popa, A. Sheffer et D. Cohen-Or. 2007, «3d collage : expressive non-realistic modeling», dans *NPAR '07 : Proceedings of the 5th international symposium on Non-photorealistic animation and rendering*, ACM, New York, NY, USA, ISBN 978-1-59593-624-0, pp. 7–14, doi :http://doi.acm.org/10.1145/1274871.1274873. URL http://www.cs.tau.ac.il/~galran/papers/collage/. 36

[25] Gardner, G. Y. 1984, «Simulation of natural scenes using textured quadric surfaces», *SIGGRAPH Comput. Graph.*, vol. 18, no. 3, doi :http://doi.acm.org/10.1145/964965.808572, pp. 11–20, ISSN 0097-8930. 42

[26] Ghazanfarpour, D. et J.-M. Dischler. 1995, «Spectral analysis for automatic 3d texture generation», *Computers & Graphics*, vol. 19, no. 3, pp. 413–422. URL http://www.msi.unilim.fr/basilic/Publications/1995/GD95. 43

[27] Glassner, A. 2005, «Speed limit 55», *IEEE Computer Graphics and Applications*, vol. 25, no. 2, doi :http://doi.ieeecomputersociety.org/10.1109/MCG.2005.45, pp. 96–106, ISSN 0272-1716. 11

[28] Gooch, B. et A. Gooch. 2001, *Non-Photorealistic Rendering*, AK Peters Ltd. URL http://www.cs.northwestern.edu/~ago820/book.html, iSBN : 1-56881-133-0. 1, 3

[29] Grundland, M., R. Vohra, G. P. Williams et N. A. Dodgson. 2006, «Cross dissolve without cross fade : preserving contrast, color and salience in image compositing», dans *Computer Graphics Forum*, vol. 25. URL http://www.cl.cam.ac.uk/research/rainbow/projects/compositing/. 29

[30] Han, C., E. Risser, R. Ramamoorthi et E. Grinspun. 2008, «Multiscale texture synthesis», *ACM Trans. Graph.* URL http://www1.cs.columbia.edu/~charhan/, to appear. 40

[31] Hays, J. et I. Essa. 2004, «Image and video based painterly animation», dans *NPAR '04 : Proceedings of the 3rd international symposium on Non-photorealistic animation and rendering*, ACM, New York, NY, USA, ISBN 1-58113-887-3, pp. 113–120, doi :http://doi.acm.org/10.1145/987657.987676. 5

[32] Heeger, D. et J. Bergen. 1995, «Pyramid-based texture analysis/synthesis», *Image Processing, 1995. Proceedings., International Conference on*, vol. 3, doi :10.1109/ICIP.1995.537718, pp. 648–651 vol.3. 30, 43

[33] Heeger, D. J. et J. R. Bergen. 1995, «Pyramid-based texture analysis/synthesis», dans *SIGGRAPH '95 : Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, ISBN 0-89791-701-4, pp. 229–238, doi :http://doi.acm.org/10.1145/218380.218446. 43, 46

[34] Hertzmann, A. 2002, «Fast paint texture», dans *NPAR '02 : Proceedings of the 2nd international symposium on Non-photorealistic animation and rendering*, ACM, New York, NY, USA, ISBN 1-58113-494-0, pp. 91–ff, doi :http://doi.acm.org/10.1145/508530.508546. 39

[35] Hertzmann, A., C. E. Jacobs, N. Oliver, B. Curless et D. H. Salesin. 2001, «Image analogies», dans *Proceedings of ACM SIGGRAPH 2001*, Computer Graphics Proceedings, Annual Conference Series, pp. 327–340, doi :http://doi.acm.org/10.1145/383259.383295. URL http://mrl.nyu.edu/projects/image-analogies/. 47

[36] Hertzmann, A. et K. Perlin. 2000, «Painterly rendering for video and interaction», dans *NPAR '00 : Proceedings of the 1st international symposium on Non-photorealistic animation and rendering*, ACM, New York, NY, USA, ISBN 1-58113-277-8, pp. 7–12, doi :http://doi.acm.org/10.1145/340916.340917. 5

[37] Hewgill, A. et B. J. Ross. 2004, «Procedural 3d texture synthesis using genetic programming», *Computers and Graphics*, vol. 28, no. 4, pp. 569–584. URL http://www.cosc.brocku.ca/~bross/research/HewgillRoss04.pdf. 42

[38] Ingram, S. et P. Bhat. 2004, «Automatic collage using texture synthesis», dans *Proceedings of 4th International Symposium on Smart Graphics*, vol. 3031, édité par A. Butz, A. Krüger et P. Olivier, pp. 140–145, doi :http://dx.doi.org/10.1007/b97744. URL http://springerlink.metapress.com/link.asp?id=wtfqt15r355x03mn. 36

[39] Jagnow, R., J. Dorsey et H. Rushmeier. 2004, «Stereological techniques for solid textures», *ACM Trans. Graph.*, vol. 23, no. 3, doi :http://doi.acm.org/10.1145/1015706.1015724, pp. 329–335, ISSN 0730-0301. URL http://www.robjagnow.com/research/. 44

[40] Kaplan, M. et E. Cohen. 2005, «A generative model for dynamic canvas motion», dans *Proceedings of the First Eurographics Workshop on Computational Aesthetics in Graphics, Visualization and Imaging 2005 (May 18–20, 2005, Girona, Spain)*, édité par L. Neumann, M. S. Casasayas, B. Gooch et W. Purgathofer, Eurographics Association, Aire-la-Ville, Switzerland, pp. 49–56, doi :http://dx.doi.org/10.2312/COMPAESTH/COMPAESTH05/049-056. 3

[41] Kaplan, M., B. Gooch et E. Cohen. 2000, «Interactive artistic rendering», dans *NPAR '00 : Proceedings of the 1st international symposium on Non-photorealistic animation and rendering*, ACM, New York, NY, USA, ISBN 1-58113-277-8, pp. 67–74, doi :http://doi.acm.org/10.1145/340916.340925. 3

[42] Klein, A. W., W. W. Li, M. M. Kazhdan, W. T. Correa, A. Finkelstein et T. A. Funkhouser. 2000, «Non-photorealistic virtual environments», dans *Proceedings of ACM SIGGRAPH 2000*, pp. 527–534. URL http://www.cs.princeton.edu/gfx/proj/nprve/. 6

[43] Kolliopoulos, A., J. M. Wang et A. Hertzmann. 2006, «Segmentation-based 3d artistic rendering», dans *Rendering Techniques 2006, Proceedings of the 17th Eurographics Symposium on Rendering (EGSR 2006, June 26–28, 2006, Nicosa, Cyprus)*, pp. 361–370, doi :http://dx.doi.org/10.2312/EGWR/EGSR06/361-370. URL http://www.dgp.toronto.edu/~alexk/segegsr.html. 8

[44] Kopf, J., D. Cohen-Or, O. Deussen et D. Lischinski. 2006, «Recursive wang tiles for real-time blue noise», *ACM Trans. Graph.*, vol. 25, no. 3, doi :http://doi.acm.org/10.1145/1141911.1141916, pp. 509–518, ISSN 0730-0301. URL http://johanneskopf.de/publications/blue_noise/. 4, 38, 41

[45] Kopf, J., C.-W. Fu, D. Cohen-Or, O. Deussen, D. Lischinski et T.-T. Wong. 2007, «Solid texture synthesis from 2d exemplars», *ACM Trans. Graph.*, vol. 26, no. 3, doi :http://doi.acm.org/10.1145/1276377.1276380, p. 2, ISSN 0730-0301. URL http://www.johanneskopf.de/publications/solid/index.php. 45, 46

[46] Kwatra, V. 2005, *Example-based Rendering of Textural Phenomena*, Ph.D. thesis, Georgia Institute of Technology. URL http://hdl.handle.net/1853/7214. 45

[47] Kwatra, V., I. Essa, A. Bobick et N. Kwatra. 2005, «Texture optimization for example-based synthesis», *ACM Transactions on Graphics, SIGGRAPH 2005*, vol. 24, doi :10.1007/s00371-006-0078-3, pp. 795–802. URL http://www.cc.gatech.edu/cpl/projects/textureoptimization/. 46

[48] Kwatra, V., A. Schödl, I. Essa, G. Turk et A. Bobick. 2003, «Graphcut textures : image and video synthesis using graph cuts», *ACM Trans. Graph.*, vol. 22, no. 3, doi :http://doi.acm.org/10.1145/882262.882264, pp. 277–286, ISSN 0730-0301. URL http://www.cc.gatech.edu/cpl/projects/graphcuttextures/. 45

[49] Lagae, A. et P. Dutré. 2005, «A procedural object distribution function», *ACM Transactions on Graphics*, vol. 24, no. 4, doi :http://doi.acm.org/10.1145/1095878.1095888, pp. 1442–1461, ISSN 0730-0301. URL http://www.cs.kuleuven.be/~ares/. 4, 41

[50] Lake, A., C. Marshall, M. Harris et M. Blackstein. 2000, «Stylized rendering techniques for scalable real-time 3d animation», dans *NPAR '00 : Proceedings of the 1st international symposium on Non-photorealistic animation and rendering*, ACM, New York, NY, USA, ISBN 1-58113-277-8, pp. 13–20, doi :http://doi.acm.org/10.1145/340916.340918. 35

[51] Lee, S. et Y. Liu. 2005, «Psu near-regular texture database», URL http://vivid.cse.psu.edu/texturedb/gallery/. 17

[52] Lefebvre, L. et P. Poulin. 2000, «Analysis and synthesis of structural textures», dans *Graphics Interface*, pp. 77–86. URL citeseer.ist.psu.edu/lefebvre00analysis.html. 42

[53] Lefebvre, S. et H. Hoppe. 2005, «Parallel controllable texture synthesis», *ACM Trans. Graph.*, vol. 24, no. 3, doi :http://doi.acm.org/10.1145/1073204.1073261, pp. 777–786, ISSN 0730-0301. 45

[54] Lefebvre, S. et H. Hoppe. 2006, «Appearance-space texture synthesis», dans *SIGGRAPH '06 : ACM SIGGRAPH 2006 Papers*, ACM, New York, NY, USA, ISBN 1-59593-364-6, pp. 541–548, doi :http://doi.acm.org/10.1145/1179352.1141921. URL http://research.microsoft.com/projects/AppTexSyn/. 23, 40, 45, 47

[55] Lei, S. I. E. et C.-F. Chang. 2004, «Real-time rendering of watercolor effects for virtual environments», dans *Advances in Multimedia Information Processing – Proceedings of the 5th Pacific Rim Conference on Multimedia (PCM 2004, Tokyo, Japan, November 30–December 3, 2004), part III*, vol. 3333, édité par K. Aizawa, Y. Nakamura et S. Satoh, pp. 474–481, doi :http://dx.doi.org/10.1007/b104121. URL http://springerlink.metapress.com/link.asp?id=kduuvdqy2vqgyxrt. 35

[56] Liang, L., C. Liu, Y.-Q. Xu, B. Guo et H.-Y. Shum. 2001, «Real-time texture synthesis by patch-based sampling», *ACM Trans. Graph.*, vol. 20, no. 3, doi :http://doi.acm.org/10.1145/501786.501787, pp. 127–150, ISSN 0730-0301. URL http://people.csail.mit.edu/celiu/Patch-basedthesis/Index.htm. 45, 47

[57] Lischinski, D. 1994, «Incremental delaunay triangulation», *Graphics gems IV*, pp. 47–59. 24

[58] Litwinowicz, P. 1997, «Processing images and video for an impressionist effect», dans *SIGGRAPH '97 : Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, ISBN 0-89791-896-7, pp. 407–414, doi : http://doi.acm.org/10.1145/258734.258893. 5

[59] Long, H., W. K. Leow et F. K. Chua. 2000, «Perceptual texture space for content-based image retrieval», dans *Proc. Int. Conf. on Multimedia Modeling*, pp. 167–180. URL citeseer.ist.psu.edu/long00perceptual.html. 18

[60] Luft, T. et O. Deussen. 2006, «Real-time watercolor illustrations of plants using a blurred depth test», dans *NPAR '06 : Proceedings of the 4th international symposium on Non-photorealistic animation and rendering*, ACM, New York, NY, USA, ISBN 1-59593-357-3, pp. 11–20, doi :http://doi.acm.org/10.1145/1124728.1124732. 35

[61] Lum, E. et K.-L. Ma. 2001, «Non-photorealistic rendering using watercolor inspired textures and illumination», *Computer Graphics and Applications, 2001. Proceedings. Ninth Pacific Conference on*, doi : 10.1109/PCCGA.2001.962888, pp. 322–330. 35

[62] Matusik, W., M. Zwicker et F. Durand. 2005, «Texture design using a simplicial complex of morphable textures», *ACM Transactions on Graphics*, vol. 24, no. 3, doi :http://doi.acm.org/10.1145/1073204.1073262, pp. 787–794, ISSN 0730-0301. URL http://people.csail.mit.edu/wojciech/TextureDesign/index.html. 24, 30

[63] Meier, B. J. 1996, «Painterly rendering for animation», dans *SIGGRAPH '96 : Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, ISBN 0-89791-746-4, pp. 477–484, doi :http://doi.acm.org/10.1145/237170.237288. 3, 4

[64] Mount, D. et S. Arya. 1997, «Ann : A library for approximate nearest neighbor searching», URL http://www.cs.umd.edu/~mount/ANN/. 24, 47

[65] Nehab, D. et P. Shilane. 2004, «Stratified point sampling of 3D models», dans *Eurographics Symposium on Point-Based Graphics*, pp. 49–56. 4

[66] Neyret, F. 2003, «Advected textures», dans *ACM-SIGGRAPH/EG Symposium on Computer Animation (SCA)*. URL http://www-evasion.imag.fr/Publications/2003/Ney03. 8

[67] Ostromoukhov, V. 1999, «Digital facial engraving», dans *SIGGRAPH '99 : Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, ISBN 0-201-48560-5, pp. 417–424, doi :http://doi.acm.org/10.1145/311535.311604. URL http://diwww.epfl.ch/w3lsp/publications/microstructureimaging/dfe.html. 38

[68] Pastor, O. M., B. Freudenberg et T. Strothotte. 2003, «Real-time animated stippling», *IEEE Comput. Graph. Appl.*, vol. 23, no. 4, doi :http://dx.doi.org/10.1109/MCG.2003.1210866, pp. 62–68, ISSN 0272-1716. 4

[69] Peachey, D. R. 1985, «Solid texturing of complex surfaces», dans *SIGGRAPH '85 : Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, ISBN 0-89791-166-0, pp. 279–286, doi :http://doi.acm.org/10.1145/325334.325246. 36, 42

[70] Pérez, P., M. Gangnet et A. Blake. 2003, «Poisson image editing», *ACM Trans. Graph.*, vol. 22, no. 3, doi : http://doi.acm.org/10.1145/882262.882269, pp. 313–318, ISSN 0730-0301. 29

[71] Perlin, K. 1985, «An image synthesizer», *SIGGRAPH Comput. Graph.*, vol. 19, no. 3, doi :http://doi.acm.org/10.1145/325165.325247, pp. 287–296, ISSN 0097-8930. 42

[72] Porter, T. et T. Duff. 1984, «Compositing digital images», *SIGGRAPH Comput. Graph.*, vol. 18, no. 3, doi : http://doi.acm.org/10.1145/964965.808606, pp. 253–259, ISSN 0097-8930. 29

[73] Praun, E., A. Finkelstein et H. Hoppe. 2000, «Lapped textures», dans *SIGGRAPH '00 : Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, ISBN 1-58113-208-5, pp. 465–470, doi :http://doi.acm.org/10.1145/344779.344987. URL http://www.cs.princeton.edu/gfx/proj/lapped_tex/. 7, 45

[74] Praun, E., H. Hoppe, M. Webb et A. Finkelstein. 2001, «Real-time hatching», dans *SIGGRAPH 2001, Computer Graphics Proceedings*, édité par E. Fiume, pp. 579–584. URL citeseer.ist.psu.edu/article/praun01realtime.htmlhttp://www.cs.princeton.edu/gfx/proj/hatching/. 6, 38

[75] Purwins, H. 2005, *Profiles of Pitch Classes Circularity of Relative Pitch and Key Experiments, Models, Computational Music Analysis, and Perspectives*, Ph.D. thesis, Elektrotechnik und Informatik der Technischen Universitt Berlin. 11, 20

[76] Qin, X. et Y.-H. Yang. 2002, «Estimating parameters for procedural texturing by genetic algorithms», *Graph. Models*, vol. 64, no. 1, doi :http://dx.doi.org/10.1006/gmod.2002.0565, pp. 19–39, ISSN 1524-0703. 42

[77] Qin, X. et Y.-H. Yang. 2007, «Aura 3d textures», *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 2, doi :http://doi.ieeecomputersociety.org/10.1109/TVCG.2007.31, pp. 379–389, ISSN 1077-2626. 44, 46

[78] Rao, A. R. et G. L. Lohse. 1993, «Identifying high level features of texture perception», *CVGIP : Graph. Models Image Process.*, vol. 55, no. 3, doi :http://dx.doi.org/10.1006/cgip.1993.1016, pp. 218–233, ISSN 1049-9652. 17

[79] Rao, A. R. et G. L. Lohse. 1993, «Towards a texture naming system : identifying relevant dimensions of texture», dans *VIS '93 : Proceedings of the 4th conference on Visualization '93*, ISBN 0-8186-3940-7, pp. 220–227, doi :10.1016/0042-6989(95)00202-2. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=398872. 17

[80] Risset, J.-C. 1986, «Pitch and rhythm paradoxes : Comments on "auditory paradox based on fractal waveform"», *The Journal of the Acoustical Society of America*, vol. 80, no. 3, doi :10.1121/1.393919, pp. 961–962. 11

[81] Secord, A. 2002, «Weighted voronoi stippling», dans *NPAR '02 : Proceedings of the 2nd international symposium on Non-photorealistic animation and rendering*, ACM, New York, NY, USA, ISBN 1-58113-494-0, pp. 37–43, doi :http://doi.acm.org/10.1145/508530.508537. URL http://mrl.nyu.edu/~ajsecord/stipples.html. 4, 38

[82] Shepard, R. N. 1964, «Circularity in judgments of relative pitch», *The Journal of the Acoustical Society of America*, vol. 36, no. 12, doi :10.1121/1.1919362, pp. 2346–2353. 10, 20

[83] Shi, J. et J. Malik. 2000, «Normalized cuts and image segmentation», *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905. URL http://techreports.lib.berkeley.edu/accessPages/CSD-97-940.html. 8

[84] Strothotte, T. et S. Schlechtweg. 2002, *Non-Photorealistic Computer Graphics : Modeling, Rendering and Animation*, Morgan Kaufmann, ISBN 1558607870. URL http://isgwww.cs.uni-magdeburg.de/pub/books/npr/. 1, 3

[85] Takayama, K., M. Okabe, T. Ijiri et T. Igarashi. 2008, «Lapped solid textures : Filling a model with anisotropic textures», *ACM Trans. Graph.* URL http://www-ui.is.s.u-tokyo.ac.jp/s/index.html, to appear. 45

[86] Tamura, H., T. Mori et T. Yamawaki. 1978, «Textural features corresponding to visual perception», *IEEE. TRANSACTIONS ON. SYSTEMS. MAN,. AND. CYBERNETICS*, vol. 8, pp. 460–473. URL http://ieeexplore.ieee.org/iel5/21/4309992/04309999.pdf?arnumber=4309999. 17

[87] Tateosian, L. G., C. G. Healey et J. T. Enns. 2007, «Engaging viewers through nonphotorealistic visualizations», dans *NPAR '07 : Proceedings of the 5th international symposium on Non-photorealistic animation and rendering*, ACM, New York, NY, USA, ISBN 978-1-59593-624-0, pp. 93–102, doi :http://doi.acm.org/10.1145/1274871.1274886. 39

[88] Tuceryan, M. et A. K. Jain. 1993, «Texture analysis», *Handbook of pattern recognition & computer vision*, pp. 235–276. 15

[89] Turk, G. 2001, «Texture synthesis on surfaces», dans *SIGGRAPH '01 : Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, ISBN 1-58113-374-X, pp. 347–354, doi :http://doi.acm.org/10.1145/383259.383297. URL http://www.cc.gatech.edu/~turk/texture_surfaces/texture.html. 45

[90] Ulichney, R. 1987, *Digital halftoning*, MIT Press, Cambridge, MA, USA, ISBN 0-262-21009-6. 3

[91] Van Laerhoven, T., J. Liesenborgs et F. Van Reeth. 2004, «Real-time watercolor painting on a distributed paper model», *Computer Graphics International, 2004. Proceedings*, doi :10.1109/CGI.2004.1309281, pp. 640–643, ISSN 1530-1052. 35

[92] Vanderhaeghe, D., P. Barla, J. Thollot et F. Sillion. 2007, «Dynamic point distribution for stroke-based rendering», dans *Rendering Techniques 2007 (Proceedings of the Eurographics Symposium on Rendering)*, pp. 139–146. URL http://artis.imag.fr/Publications/2007/VBTS07a. 5

[93] Vanderhaeghe, D., P. Barla, J. Thollot et F. Sillion. 2007, «Rendu peinture interactif et controlable de scènes 3d», *Revue Electronique Francophone d'Informatique Graphique*, vol. 1, no. 1, pp. 53–62. URL http://artis.imag.fr/Publications/2007/VBTS07. 3

[94] Vergne, R., P. Barla, X. Granier et C. Schlick. 2008, «Apparent relief : a shape descriptor for stylized shading», dans *NPAR '08 : Proceedings of the 6th international symposium on Non-photorealistic animation and rendering*, ACM. URL http://iparla.labri.fr/publications/2008/VBGS08. 37

[95] Wang, J., Y. Xu, H.-Y. Shum et M. F. Cohen. 2004, «Video tooning», *ACM Trans. Graph.*, vol. 23, no. 3, doi :http://doi.acm.org/10.1145/1015706.1015763, pp. 574–583, ISSN 0730-0301. 8

[96] Wei, L.-Y. 2002, *Texture synthesis by fixed neighborhood searching*, Ph.D. thesis, Stanford University, Stanford, CA, USA. URL http://graphics.stanford.edu/papers/liyiwei_thesis/, adviser-Marc Levoy. 44, 46

[97] Wei, L.-Y. et M. Levoy. 2000, «Fast texture synthesis using tree-structured vector quantization», dans *SIGGRAPH '00 : Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, ISBN 1-58113-208-5, pp. 479–488, doi :http://doi.acm.org/10.1145/344779.345009. URL http://graphics.stanford.edu/papers/texture-synthesis-sig00/. 45

[98] Wexler, Y., E. Shechtman et M. Irani. 2007, «Space-time completion of video», *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 3, doi :http://doi.ieeecomputersociety.org/10.1109/TPAMI.2007.60, pp. 463–476, ISSN 0162-8828. URL http://www.wisdom.weizmann.ac.il/~vision/VideoCompletion.html. 45, 47

[99] Wu, Q. et Y. Yu. 2004, «Feature matching and deformation for texture synthesis», *ACM Trans. Graph.*, vol. 23, no. 3, doi :http://doi.acm.org/10.1145/1015706.1015730, pp. 364–367, ISSN 0730-0301. 23, 47